Basic Phrase Based Decoding

1.0

Generated by Doxygen 1.8.10

Tue Mar 15 2016 14:17:10

Contents

1 README **1**

2 Todo List **19**

3 Namespace Index **23**

3.1 Namespace List 23

4 Hierarchical Index **25**

4.1 Class Hierarchy 25

5 Class Index **29**

5.1 Class List 29

6 File Index **33**

6.1 File List 33

7 Namespace Documentation **37**

7.1 uva Namespace Reference 37

7.2 uva::smt Namespace Reference 37

7.3 uva::smt::bpbd Namespace Reference 37

7.4 uva::smt::bpbd::client Namespace Reference 37

7.4.1 Typedef Documentation 38

7.4.1.1 trans_job_ptr 38

7.4.2 Function Documentation 38

7.4.2.1 operator<<(ostream &os, const trans_job_status &status) 38

7.5 uva::smt::bpbd::common Namespace Reference 38

7.5.1 Function Documentation 38

7.5.1.1 get_float(INI<> &ini, string section, string key) 38

7.5.1.2 get_integer(INI<> &ini, string section, string key) 38

7.5.1.3 get_string(INI<> &ini, string section, string key) 39

7.6 uva::smt::bpbd::common::messaging Namespace Reference 39

7.6.1 Typedef Documentation 39

7.6.1.1 job_id_type 39

7.6.1.2 session_id_type 39

7.6.1.3 trans_job_request_ptr . . . 39
7.6.1.4 trans_job_response_ptr . . . 39
7.6.2 Function Documentation . . . 40
7.6.2.1 operator<<(ostream &os, const trans_job_code &code) . . . 40
7.7 uva::smt::bpbd::common::messaging::job_id Namespace Reference . . . 41
7.8 uva::smt::bpbd::common::messaging::session_id Namespace Reference . . . 41
7.9 uva::smt::bpbd::server Namespace Reference . . . 41
7.9.1 Typedef Documentation . . . 42
7.9.1.1 phrase_length . . . 42
7.9.1.2 phrase_uid . . . 42
7.9.1.3 prob_weight . . . 42
7.9.1.4 task_id_type . . . 42
7.9.1.5 trans_job_ptr . . . 42
7.9.1.6 trans_task_ptr . . . 42
7.9.1.7 word_uid . . . 42
7.9.2 Function Documentation . . . 43
7.9.2.1 begins_with(const string &str, const string &prefix) . . . 43
7.9.2.2 get_float_value(const string &str, const string &prefix) . . . 44
7.9.2.3 get_int_value(const string &str, const string &prefix) . . . 44
7.9.2.4 get_string_value(const string &str, const string &prefix) . . . 44
7.9.2.5 perform_command_loop(server_parameters &params, translation_server &server, thread &server_thread) . . . 44
7.9.2.6 print_server_commands() . . . 45
7.9.2.7 print_the_prompt() . . . 45
7.9.2.8 process_input_cmd(server_parameters &params, translation_server &server, thread &server_thread, char command[CMD_BUFF_SIZE]) . . . 45
7.9.2.9 set_decoder_params(const string &cmd, de_parameters &de_params) . . . 45
7.9.2.10 set_log_level(const string &cmd, const string &prefix) . . . 45
7.9.2.11 set_num_threads(server_parameters &params, translation_server &server, const string &cmd, const string &prefix) . . . 46
7.9.2.12 stop(translation_server &server, thread &server_thread) . . . 46
7.10 uva::smt::bpbd::server::common Namespace Reference . . . 46
7.11 uva::smt::bpbd::server::common::models Namespace Reference . . . 46
7.12 uva::smt::bpbd::server::decoder Namespace Reference . . . 46
7.12.1 Typedef Documentation . . . 47
7.12.1.1 de_parameters . . . 47
7.13 uva::smt::bpbd::server::decoder::sentence Namespace Reference . . . 47
7.13.1 Typedef Documentation . . . 47
7.13.1.1 sentence_data_map . . . 47
7.14 uva::smt::bpbd::server::decoder::stack Namespace Reference . . . 47
7.14.1 Typedef Documentation . . . 47

7.14.1.1 add_new_state_function . . . 47
7.14.1.2 stack_level_ptr . . . 48
7.14.1.3 stack_state . . . 48
7.14.1.4 stack_state_ptr . . . 48
7.15 uva::smt::bpbd::server::lm Namespace Reference . . . 48
7.15.1 Typedef Documentation . . . 51
7.15.1.1 lm_builder_type . . . 51
7.15.1.2 lm_model_reader . . . 51
7.15.1.3 lm_model_type . . . 51
7.15.1.4 lm_word_index . . . 51
7.15.1.5 TC2DHybridTrieBasic . . . 51
7.15.1.6 TC2DHybridTrieCount . . . 51
7.15.1.7 TC2DHybridTrieHashing . . . 51
7.15.1.8 TC2DHybridTrieOptBasic . . . 51
7.15.1.9 TC2DHybridTrieOptCount . . . 51
7.15.1.10 TC2DMapTrieBasic . . . 51
7.15.1.11 TC2DMapTrieCount . . . 51
7.15.1.12 TC2DMapTrieHashing . . . 52
7.15.1.13 TC2DMapTrieOptBasic . . . 52
7.15.1.14 TC2DMapTrieOptCount . . . 52
7.15.1.15 TC2WArrayTrieBasic . . . 52
7.15.1.16 TC2WArrayTrieCount . . . 52
7.15.1.17 TC2WArrayTrieHashing . . . 52
7.15.1.18 TC2WArrayTrieOptBasic . . . 52
7.15.1.19 TC2WArrayTrieOptCount . . . 52
7.15.1.20 TG2DMapTrieBasic . . . 52
7.15.1.21 TG2DMapTrieCount . . . 52
7.15.1.22 TG2DMapTrieHashing . . . 52
7.15.1.23 TG2DMapTrieOptBasic . . . 53
7.15.1.24 TG2DMapTrieOptCount . . . 53
7.15.1.25 TH2DMapTrieBasic . . . 53
7.15.1.26 TH2DMapTrieCount . . . 53
7.15.1.27 TH2DMapTrieHashing . . . 53
7.15.1.28 TH2DMapTrieOptBasic . . . 53
7.15.1.29 TH2DMapTrieOptCount . . . 53
7.15.1.30 TStorageMap . . . 53
7.15.1.31 TStorageMapAllocator . . . 53
7.15.1.32 TStorageMapEntry . . . 53
7.15.1.33 TStorageUnsignedMap . . . 53
7.15.1.34 TW2CArrayTrieBasic . . . 54

7.15.1.35 TW2CArrayTrieCount . . . 54
7.15.1.36 TW2CArrayTrieHashing . . . 54
7.15.1.37 TW2CArrayTrieOptBasic . . . 54
7.15.1.38 TW2CArrayTrieOptCount . . . 54
7.15.1.39 TW2CHybridTrieBasic . . . 54
7.15.1.40 TW2CHybridTrieCount . . . 54
7.15.1.41 TW2CHybridTrieHashing . . . 54
7.15.1.42 TW2CHybridTrieOptBasic . . . 54
7.15.1.43 TW2CHybridTrieOptCount . . . 54
7.15.2 Enumeration Type Documentation . . . 55
7.15.2.1 MGramStatusEnum . . . 55
7.15.3 Function Documentation . . . 55
7.15.3.1 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(w2c_array_trie, basic_word_index) . . . 55
7.15.3.2 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(w2c_array_trie, counting_word_index) . . . 55
7.15.3.3 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(w2c_array_trie, hashing_word_index) . . . 55
7.15.3.4 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(w2c_array_trie, basic_optimizing_word_index) . . . 55
7.15.3.5 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(w2c_array_trie, counting_optimizing_word_index) . . . 55
7.15.3.6 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(c2w_array_trie, basic_word_index) . . . 55
7.15.3.7 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(c2w_array_trie, counting_word_index) . . . 55
7.15.3.8 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(c2w_array_trie, hashing_word_index) . . . 55
7.15.3.9 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(c2w_array_trie, basic_optimizing_word_index) . . . 55
7.15.3.10 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(c2w_array_trie, counting_optimizing_word_index) . . . 55
7.15.3.11 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(w2c_hybrid_trie, basic_word_index) . . . 55
7.15.3.12 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(w2c_hybrid_trie, counting_word_index) . . . 56
7.15.3.13 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(w2c_hybrid_trie, hashing_word_index) . . . 56
7.15.3.14 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(w2c_hybrid_trie, basic_optimizing_word_index) . . . 56
7.15.3.15 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(w2c_hybrid_trie, counting_optimizing_word_index) . . . 56
7.15.3.16 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(c2d_map_trie, basic_word_index) . . . 56

7.15.3.17 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(c2d_map_trie, counting_word_index) . . . 56
7.15.3.18 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(c2d_map_trie, hashing_word_index) . . . 56
7.15.3.19 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(c2d_map_trie, basic_optimizing_word_index) . . . 56
7.15.3.20 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(c2d_map_trie, counting_optimizing_word_index) . . . 56
7.15.3.21 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(c2d_hybrid_trie, basic_word_index) . . . 56
7.15.3.22 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(c2d_hybrid_trie, counting_word_index) . . . 56
7.15.3.23 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(c2d_hybrid_trie, hashing_word_index) . . . 56
7.15.3.24 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(c2d_hybrid_trie, basic_optimizing_word_index) . . . 56
7.15.3.25 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(c2d_hybrid_trie, counting_optimizing_word_index) . . . 56
7.15.3.26 INSTANTIATE_TRIE_TEMPLATE_TYPE(h2d_map_trie, basic_word_index) . . 56
7.15.3.27 INSTANTIATE_TRIE_TEMPLATE_TYPE(h2d_map_trie, counting_word_index) . 56
7.15.3.28 INSTANTIATE_TRIE_TEMPLATE_TYPE(h2d_map_trie, hashing_word_index) . 56
7.15.3.29 INSTANTIATE_TRIE_TEMPLATE_TYPE(h2d_map_trie, basic_optimizing_word_index) . . . 56
7.15.3.30 INSTANTIATE_TRIE_TEMPLATE_TYPE(h2d_map_trie, counting_optimizing_word_index) . . . 56
7.15.3.31 INSTANTIATE_TRIE_TEMPLATE_TYPE(g2d_map_trie, basic_word_index) . . 56
7.15.3.32 INSTANTIATE_TRIE_TEMPLATE_TYPE(g2d_map_trie, counting_word_index) . 57
7.15.3.33 INSTANTIATE_TRIE_TEMPLATE_TYPE(g2d_map_trie, hashing_word_index) . 57
7.15.3.34 INSTANTIATE_TRIE_TEMPLATE_TYPE(g2d_map_trie, basic_optimizing_word_index) . . . 57
7.15.3.35 INSTANTIATE_TRIE_TEMPLATE_TYPE(g2d_map_trie, counting_optimizing_word_index) . . . 57
7.15.3.36 operator<<(ostream &stream, const m_gram_query &query) . . . 57
7.15.4 Variable Documentation . . . 57
7.15.4.1 __attribute__ . . . 57
7.15.4.2 DEF_UNK_WORD_LOG_PROB_WEIGHT . . . 57
7.16 uva::smt::bpbd::server::lm::__C2DHybridTrie Namespace Reference . . . 57
7.17 uva::smt::bpbd::server::lm::__C2DMapTrie Namespace Reference . . . 57
7.18 uva::smt::bpbd::server::lm::__C2WArrayTrie Namespace Reference . . . 57
7.18.1 Function Documentation . . . 58
7.18.1.1 compare(const TCtxIdProbData &one, const TCtxIdProbData &two) . . . 58
7.18.1.2 operator<(const TWordIdPBData &one, const TWordIdPBData &two) . . . 59
7.18.1.3 operator<(const TCtxIdProbData &one, const TCtxIdProbData &two) . . . 59
7.18.1.4 operator==(const TCtxIdProbData &one, const TCtxIdProbData &two) . . . 59

7.18.1.5 operator>(const TCtxIdProbData &one, const TCtxIdProbData &two) 59
7.19 uva::smt::bpbd::server::lm::__executor Namespace Reference 59
7.20 uva::smt::bpbd::server::lm::__G2DMapTrie Namespace Reference 59
7.21 uva::smt::bpbd::server::lm::__H2DMapTrie Namespace Reference 60
7.22 uva::smt::bpbd::server::lm::__LayeredTrieBase Namespace Reference 60
7.22.1 Function Documentation 60
7.22.1.1 get_context_id(TrieType &trie, const model_m_gram &gram, TLongId &ctx_id) . . 60
7.22.1.2 search_m_gram_ctx_id(const TrieType &trie, const word_uid *const word_ids, TLongId &prev_ctx_id, TLongId &ctx_id) 60
7.23 uva::smt::bpbd::server::lm::__W2CArrayTrie Namespace Reference 61
7.23.1 Typedef Documentation 61
7.23.1.1 T_M_GramData 61
7.23.1.2 T_N_GramData 61
7.23.2 Function Documentation 61
7.23.2.1 operator<(const T_M_GramData &one, const T_M_GramData &two) 61
7.23.2.2 operator<(const T_N_GramData &one, const T_N_GramData &two) 62
7.24 uva::smt::bpbd::server::lm::__W2CHybridTrie Namespace Reference 62
7.25 uva::smt::bpbd::server::lm::arpa Namespace Reference 62
7.25.1 Function Documentation 63
7.25.1.1 INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL(M_GRAM_LEVEL_1) 63
7.25.1.2 INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL(M_GRAM_LEVEL_2) 63
7.25.1.3 INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL(M_GRAM_LEVEL_3) 63
7.25.1.4 INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL(M_GRAM_LEVEL_4) 63
7.25.1.5 INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL(M_GRAM_LEVEL_5) 63
7.25.1.6 INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL(M_GRAM_LEVEL_6) 63
7.25.1.7 INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL(M_GRAM_LEVEL_7) 63
7.25.1.8 INSTANTIATE_TRIE_BUILDER_FILE_READER(cstyle_file_reader) 63
7.25.1.9 INSTANTIATE_TRIE_BUILDER_FILE_READER(file_stream_reader) 63
7.25.1.10 INSTANTIATE_TRIE_BUILDER_FILE_READER(memory_mapped_file_reader) 63
7.26 uva::smt::bpbd::server::lm::caching Namespace Reference 63
7.27 uva::smt::bpbd::server::lm::dictionary Namespace Reference 63
7.27.1 Typedef Documentation 64
7.27.1.1 basic_optimizing_word_index 64
7.27.1.2 counting_optimizing_word_index 64
7.28 uva::smt::bpbd::server::lm::dictionary::__AWordIndex Namespace Reference 64
7.29 uva::smt::bpbd::server::lm::dictionary::__counting_word_index Namespace Reference 64
7.29.1 Function Documentation 64
7.29.1.1 operator<(const TWordInfo &one, const TWordInfo &two) 64
7.30 uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index Namespace Reference 64
7.30.1 Function Documentation 65

7.30.1.1 __attribute__((packed)) . . . 65
7.31 uva::smt::bpbd::server::lm::identifiers Namespace Reference . . . 65
7.31.1 Detailed Description . . . 65
7.31.2 Typedef Documentation . . . 65
7.31.2.1 TLongId . . . 65
7.31.2.2 TShortId . . . 65
7.32 uva::smt::bpbd::server::lm::m_grams Namespace Reference . . . 65
7.32.1 Typedef Documentation . . . 66
7.32.1.1 m_gram_payload . . . 66
7.32.2 Function Documentation . . . 66
7.32.2.1 operator<<(ostream &stream, const model_m_gram &gram) . . . 66
7.32.2.2 operator<<(ostream &stream, const query_m_gram &gram) . . . 66
7.33 uva::smt::bpbd::server::lm::m_grams::m_gram_id Namespace Reference . . . 66
7.33.1 Detailed Description . . . 67
7.33.2 Typedef Documentation . . . 67
7.33.2.1 TM_Gram_Id_Value_Ptr . . . 67
7.34 uva::smt::bpbd::server::lm::proxy Namespace Reference . . . 67
7.35 uva::smt::bpbd::server::rm Namespace Reference . . . 67
7.35.1 Typedef Documentation . . . 67
7.35.1.1 rm_builder_type . . . 67
7.35.1.2 rm_model_reader . . . 68
7.35.1.3 rm_model_type . . . 68
7.36 uva::smt::bpbd::server::rm::builders Namespace Reference . . . 68
7.37 uva::smt::bpbd::server::rm::models Namespace Reference . . . 68
7.37.1 Typedef Documentation . . . 68
7.37.1.1 rm_entry . . . 68
7.37.2 Enumeration Type Documentation . . . 68
7.37.2.1 reordering_orientation . . . 68
7.38 uva::smt::bpbd::server::rm::models::__rm_basic_model Namespace Reference . . . 69
7.39 uva::smt::bpbd::server::rm::proxy Namespace Reference . . . 69
7.40 uva::smt::bpbd::server::task_id Namespace Reference . . . 69
7.41 uva::smt::bpbd::server::tm Namespace Reference . . . 69
7.41.1 Typedef Documentation . . . 69
7.41.1.1 tm_builder_type . . . 69
7.41.1.2 tm_model_reader . . . 70
7.41.1.3 tm_model_type . . . 70
7.42 uva::smt::bpbd::server::tm::builders Namespace Reference . . . 70
7.42.1 Typedef Documentation . . . 70
7.42.1.1 sizes_map . . . 70
7.43 uva::smt::bpbd::server::tm::models Namespace Reference . . . 70

7.43.1 Typedef Documentation . . . 71
7.43.1.1 feature_array . . . 71
7.43.1.2 tm_const_source_entry . . . 71
7.43.1.3 tm_const_source_entry_ptr . . . 71
7.43.1.4 tm_const_target_entry . . . 71
7.43.1.5 tm_source_entry_ptr . . . 71
7.43.1.6 tm_target_entry . . . 71
7.44 uva::smt::bpbd::server::tm::models::__tm_basic_model Namespace Reference . . . 71
7.45 uva::smt::bpbd::server::tm::proxy Namespace Reference . . . 71
7.46 uva::utils Namespace Reference . . . 71
7.47 uva::utils::containers Namespace Reference . . . 72
7.47.1 Typedef Documentation . . . 72
7.47.1.1 TCapacityIncFunct . . . 72
7.47.2 Enumeration Type Documentation . . . 73
7.47.2.1 mem_inc_types_enum . . . 73
7.47.3 Function Documentation . . . 73
7.47.3.1 get_mem_incr_strat(const mem_inc_types_enum stype, const size_t min_mem↩ _inc, const size_t mem_inc_factor) . . . 73
7.47.4 Variable Documentation . . . 73
7.47.4.1 _memIncTypesEnumStr . . . 73
7.48 uva::utils::containers::alloc Namespace Reference . . . 74
7.48.1 Function Documentation . . . 74
7.48.1.1 allocate_container(TContaner **ppContainer, TAllocator **ppAllocator, const size_t numEntries, const string ctName, const float factor=UNORDERED_M↩ AP_MEMORY_FACTOR) . . . 74
7.48.1.2 deallocate_container(TContaner **ppContainer, TAllocator **ppAllocator) . . . 74
7.48.1.3 operator"!=(const greedy_memory_allocator< T > &, const greedy_memory_↩ allocator< U > &) . . . 75
7.48.1.4 operator"!=(const greedy_memory_allocator< T > &, const greedy_memory_↩ allocator< T > &) . . . 75
7.48.1.5 operator==(const greedy_memory_allocator< T > &, const greedy_memory_↩ allocator< U > &) . . . 75
7.48.1.6 operator==(const greedy_memory_allocator< T > &, const greedy_memory_↩ allocator< T > &) . . . 75
7.48.1.7 reserve_mem_unordered_map(TContaner **ppContainer, TAllocator **pp↩ Allocator, const size_t numEntries, const string ctName, const float factor=UN↩ ORDERED_MAP_MEMORY_FACTOR) . . . 75
7.49 uva::utils::containers::utils Namespace Reference . . . 75
7.49.1 Function Documentation . . . 76
7.49.1.1 is_less(const ELEM_TYPE &first, const ELEM_TYPE &second) . . . 76
7.49.1.2 my_bsearch(const ARR_ELEM_TYPE *array, INDEX_TYPE l_idx, INDEX_TY↩ PE u_idx, const KEY_TYPE key, INDEX_TYPE &mid_pos) . . . 76

7.49.1.3 my_bsearch_id(const ARR_ELEM_TYPE *array, int64_t l_idx, int64_t u_idx, const typename ARR_ELEM_TYPE::TIdType key, const ARR_ELEM_TYPE *&found_elem) . . . 77
7.49.1.4 my_bsearch_id(const ARR_ELEM_TYPE *array, int64_t l_idx, int64_t u_idx, const KEY_TYPE key, IDX_TYPE &found_pos) . . . 77
7.49.1.5 my_bsearch_wordId_ctxId(const ARR_ELEM_TYPE *array, int64_t l_idx, int64_t u_idx, const uint32_t key1, const uint32_t key2, uint32_t &found_pos) . . . 78
7.49.1.6 my_isearch_id(const ARR_ELEM_TYPE *array, int64_t l_idx, int64_t u_idx, const KEY_TYPE key, const ARR_ELEM_TYPE *&found_elem) . . . 78
7.49.1.7 my_lsearch_id(const ARR_ELEM_TYPE *array, int64_t l_idx, int64_t u_idx, const typename ARR_ELEM_TYPE::TIdType key, const ARR_ELEM_TYPE *&found_elem) . . . 79
7.49.1.8 my_sort(ELEM_TYPE *array_begin, const uint32_t array_size, typename T_IS_COMPARE_FUNC< ELEM_TYPE >::func_type is_less_func) . . . 79
7.49.1.9 my_sort(ELEM_TYPE *array_begin, const uint32_t array_size) . . . 80
7.50 uva::utils::exceptions Namespace Reference . . . 80
7.50.1 Variable Documentation . . . 80
7.50.1.1 DO_SANITY_CHECKS . . . 80
7.51 uva::utils::file Namespace Reference . . . 80
7.51.1 Function Documentation . . . 81
7.51.1.1 operator<<(ostream &output, const text_piece_reader &val) . . . 81
7.51.1.2 tokens_to_string(const text_piece_reader tokens[NUM_TOKENS], const size_t begin_idx, const size_t end_idx) . . . 81
7.52 uva::utils::hashing Namespace Reference . . . 81
7.53 uva::utils::logging Namespace Reference . . . 81
7.53.1 Enumeration Type Documentation . . . 82
7.53.1.1 debug_levels_enum . . . 82
7.53.2 Function Documentation . . . 82
7.53.2.1 operator<<(std::ostream &stream, const unsigned char &value) . . . 82
7.53.2.2 operator<<(std::ostream &stream, const signed char &value) . . . 82
7.54 uva::utils::math Namespace Reference . . . 83
7.55 uva::utils::math::bits Namespace Reference . . . 83
7.56 uva::utils::math::const_expr Namespace Reference . . . 83
7.56.1 Function Documentation . . . 83
7.56.1.1 ceil(double value) . . . 83
7.56.1.2 log2(double value, double pow=0.0) . . . 83
7.56.1.3 power(uint64_t value, uint8_t pow) . . . 83
7.57 uva::utils::math::log2 Namespace Reference . . . 83
7.58 uva::utils::monitor Namespace Reference . . . 83
7.58.1 Typedef Documentation . . . 84
7.58.1.1 TMemotyUsage . . . 84
7.58.2 Variable Documentation . . . 84
7.58.2.1 BYTES_ONE_MB . . . 84

7.59 uva::utils::text Namespace Reference . . . 84
7.59.1 Variable Documentation . . . 84
7.59.1.1 ASCII_SPACE_CHAR . . . 84
7.59.1.2 UTF8_ASCII_PUNCTUATIONS . . . 84
7.59.1.3 UTF8_ASCII_WHITESPACES . . . 84
7.59.1.4 UTF8_EMPTY_STRING . . . 84
7.59.1.5 UTF8_NEW_LINE_STRING . . . 84
7.59.1.6 UTF8_SPACE_STRING . . . 85
7.60 uva::utils::threads Namespace Reference . . . 85
7.60.1 Typedef Documentation . . . 85
7.60.1.1 a_bool_flag . . . 85
7.60.1.2 acr_bool_flag . . . 85
7.60.1.3 recursive_guard . . . 85
7.60.1.4 scoped_guard . . . 85
7.60.1.5 unique_guard . . . 85

8 Class Documentation **87**

8.1 uva::utils::file::afile_reader Class Reference . . . 87
8.1.1 Detailed Description . . . 87
8.1.2 Constructor & Destructor Documentation . . . 88
8.1.2.1 afile_reader() . . . 88
8.1.2.2 ∼afile_reader() . . . 88
8.1.3 Member Function Documentation . . . 88
8.1.3.1 close() . . . 88
8.1.3.2 get_first(text_piece_reader &out) . . . 88
8.1.3.3 get_first_line(text_piece_reader &out) . . . 88
8.1.3.4 get_first_space(text_piece_reader &out) . . . 88
8.1.3.5 get_first_tab(text_piece_reader &out) . . . 88
8.1.3.6 get_last(text_piece_reader &out) . . . 88
8.1.3.7 get_last_space(text_piece_reader &out) . . . 89
8.1.3.8 is_open() const =0 . . . 89
8.1.3.9 log_reader_type_info()=0 . . . 89
8.1.3.10 operator bool() const =0 . . . 89
8.1.3.11 reset() . . . 89
8.2 uva::smt::bpbd::server::lm::dictionary::aword_index Class Reference . . . 89
8.2.1 Detailed Description . . . 90
8.2.2 Constructor & Destructor Documentation . . . 90
8.2.2.1 ∼aword_index() . . . 90
8.2.3 Member Function Documentation . . . 90
8.2.3.1 count_word(const text_piece_reader &word, prob_weight prob) . . . 90

8.2.3.2 do_post_actions() . . . 91
8.2.3.3 do_post_word_count() . . . 91
8.2.3.4 get_number_of_words(const size_t num_words) const . . . 91
8.2.3.5 get_word_id(const text_piece_reader &token) const . . . 91
8.2.3.6 is_post_actions_needed() const . . . 91
8.2.3.7 is_word_counts_needed() const . . . 91
8.2.3.8 is_word_index_continuous() . . . 92
8.2.3.9 is_word_registering_needed() const . . . 92
8.2.3.10 register_word(const text_piece_reader &token) . . . 92
8.2.3.11 reserve(const size_t num_words) . . . 92
8.3 uva::smt::bpbd::server::lm::dictionary::basic_word_index Class Reference . . . 92
8.3.1 Detailed Description . . . 93
8.3.2 Member Typedef Documentation . . . 94
8.3.2.1 TWordIndexAllocator . . . 94
8.3.2.2 TWordIndexEntry . . . 94
8.3.2.3 TWordIndexMap . . . 94
8.3.2.4 TWordIndexMapConstIter . . . 94
8.3.3 Constructor & Destructor Documentation . . . 94
8.3.3.1 basic_word_index(const float wordIndexMemFactor) . . . 94
8.3.3.2 ~basic_word_index() . . . 94
8.3.3.3 basic_word_index(const basic_word_index &other) . . . 94
8.3.4 Member Function Documentation . . . 95
8.3.4.1 begin() . . . 95
8.3.4.2 count_word(const text_piece_reader &word, prob_weight prob) . . . 95
8.3.4.3 do_post_actions() . . . 95
8.3.4.4 do_post_word_count() . . . 95
8.3.4.5 end() . . . 95
8.3.4.6 get_number_of_words(const size_t num_words) const . . . 96
8.3.4.7 get_word_id(const text_piece_reader &token) const . . . 96
8.3.4.8 is_post_actions_needed() const . . . 96
8.3.4.9 is_word_counts_needed() const . . . 96
8.3.4.10 is_word_index_continuous() . . . 96
8.3.4.11 is_word_registering_needed() const . . . 97
8.3.4.12 register_word(const text_piece_reader &token) . . . 97
8.3.4.13 reserve(const size_t num_words) . . . 97
8.3.5 Member Data Documentation . . . 97
8.3.5.1 m_next_new_word_id . . . 97
8.3.5.2 m_word_index_alloc_ptr . . . 97
8.3.5.3 m_word_index_map_ptr . . . 97
8.3.5.4 m_word_index_mem_factor . . . 97

8.4 uva::smt::bpbd::server::lm::caching::BitmapHashCache Class Reference . . . 98
8.4.1 Detailed Description . . . 98
8.4.2 Constructor & Destructor Documentation . . . 98
8.4.2.1 BitmapHashCache() . . . 98
8.4.2.2 ∼BitmapHashCache() . . . 98
8.4.3 Member Function Documentation . . . 98
8.4.3.1 cache_m_gram_hash(const model_m_gram gram) . . . 98
8.4.3.2 is_hash_cached(uint_fast64_t key) const . . . 98
8.4.3.3 pre_allocate(const size_t num_elems, const uint8_t buckets_factor) . . . 99
8.5 uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType > Class Template Reference . . . 99
8.5.1 Detailed Description . . . 103
8.5.2 Member Function Documentation . . . 103
8.5.2.1 allocate_byte_m_gram_id(const phrase_length level, TM_Gram_Id_Value_Ptr &m_p_gram_id) . . . 103
8.5.2.2 compare(const uint8_t id_len_bytes, const TM_Gram_Id_Value_Ptr &m_p_↩ gram_id_one, const TM_Gram_Id_Value_Ptr &m_p_gram_id_two) . . . 104
8.5.2.3 compute_m_gram_id(const TWordIdType ∗word_ids, const uint8_t num_word_↩ ids, TM_Gram_Id_Value_Ptr m_p_gram_id) . . . 105
8.5.2.4 create_m_gram_id(const TWordIdType ∗word_ids, const uint8_t num_word_ids, TM_Gram_Id_Value_Ptr &m_p_gram_id) . . . 105
8.5.2.5 gram_id_byte_len_2_type(const phrase_length gram_level, uint8_t ∗len_bytes) . 106
8.5.2.6 gram_id_type_2_byte_len(uint32_t id_type) . . . 106
8.5.2.7 is_equal_m_grams_id(const uint8_t id_len_bytes, const TM_Gram_Id_Value_Ptr &one, const TM_Gram_Id_Value_Ptr &two) . . . 107
8.5.2.8 is_less_m_grams_id(const uint8_t id_len_bytes, const TM_Gram_Id_Value_Ptr &one, const TM_Gram_Id_Value_Ptr &two) . . . 108
8.5.2.9 is_less_m_grams_id(const uint8_t id_type_len_bytes, const TM_Gram_Id_↩ Value_Ptr &one, const TM_Gram_Id_Value_Ptr &two) . . . 108
8.5.2.10 is_more_m_grams_id(const uint8_t id_len_bytes, const TM_Gram_Id_Value_Ptr &one, const TM_Gram_Id_Value_Ptr &two) . . . 108
8.5.3 Member Data Documentation . . . 109
8.5.3.1 ID_TYPE_LEN_BYTES . . . 109
8.5.3.2 LEVEL_2_GRAM_TO_BYTE_LEN . . . 109
8.5.3.3 LEVEL_2_GRAM_TO_TYPE_LEN . . . 109
8.5.3.4 LEVEL_3_GRAM_TO_BYTE_LEN . . . 110
8.5.3.5 LEVEL_3_GRAM_TO_TYPE_LEN . . . 110
8.5.3.6 LEVEL_4_GRAM_TO_BYTE_LEN . . . 111
8.5.3.7 LEVEL_4_GRAM_TO_TYPE_LEN . . . 111
8.5.3.8 LEVEL_5_GRAM_TO_BYTE_LEN . . . 111
8.5.3.9 LEVEL_5_GRAM_TO_TYPE_LEN . . . 112
8.5.3.10 LEVEL_6_GRAM_TO_BYTE_LEN . . . 113
8.5.3.11 LEVEL_6_GRAM_TO_TYPE_LEN . . . 114

8.5.3.12 MAX_ID_LEN_BYTES . . . 114
8.5.3.13 NUM_BYTES_WORD_ID . . . 114
8.5.3.14 NUMBER_ID_TYPES_PER_LEVEL . . . 114
8.6 uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType > Class Template Reference . . . 115
8.6.1 Detailed Description . . . 115
8.6.2 Member Typedef Documentation . . . 116
8.6.2.1 BASE . . . 116
8.6.3 Constructor & Destructor Documentation . . . 116
8.6.3.1 c2d_hybrid_trie(WordIndexType &word_index, const float mram_mem_factor=_↩ _C2DHybridTrie::UM_M_GRAM_MEMORY_FACTOR, const float ngram_mem↩ _factor=__C2DHybridTrie::UM_N_GRAM_MEMORY_FACTOR) . . . 116
8.6.3.2 ∼c2d_hybrid_trie() . . . 116
8.6.4 Member Function Documentation . . . 116
8.6.4.1 add_m_gram(const model_m_gram &gram) . . . 116
8.6.4.2 get_ctx_id(const phrase_length level_idx, const TShortId word_id, TLongId &ctx_id) const . . . 117
8.6.4.3 get_m_gram_payload(m_gram_query &query, MGramStatusEnum &status) const . . . 117
8.6.4.4 get_n_gram_payload(m_gram_query &query, MGramStatusEnum &status) const 117
8.6.4.5 get_unigram_payload(m_gram_query &query) const . . . 117
8.6.4.6 get_unk_word_prob() const . . . 117
8.6.4.7 log_model_type_info() const . . . 118
8.6.4.8 pre_allocate(const size_t counts[LM_M_GRAM_LEVEL_MAX]) . . . 118
8.6.4.9 set_def_unk_word_prob(const prob_weight prob) . . . 118
8.7 uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType > Class Template Reference . . . 118
8.7.1 Detailed Description . . . 119
8.7.2 Member Typedef Documentation . . . 119
8.7.2.1 BASE . . . 119
8.7.3 Constructor & Destructor Documentation . . . 119
8.7.3.1 c2d_map_trie(WordIndexType &word_index, const float mgram_mem_factor=_↩ _C2DMapTrie::UM_M_GRAM_MEMORY_FACTOR, const float ngram_mem_↩ factor=__C2DMapTrie::UM_N_GRAM_MEMORY_FACTOR) . . . 119
8.7.3.2 ∼c2d_map_trie() . . . 120
8.7.4 Member Function Documentation . . . 120
8.7.4.1 add_m_gram(const model_m_gram &gram) . . . 120
8.7.4.2 get_ctx_id(const phrase_length level_idx, const TShortId word_id, TLongId &ctx_id) const . . . 120
8.7.4.3 get_m_gram_payload(m_gram_query &query, MGramStatusEnum &status) const . . . 120
8.7.4.4 get_n_gram_payload(m_gram_query &query, MGramStatusEnum &status) const 121
8.7.4.5 get_unigram_payload(m_gram_query &query) const . . . 121
8.7.4.6 get_unk_word_prob() const . . . 121

8.7.4.7 log_model_type_info() const . . . 121
8.7.4.8 pre_allocate(const size_t counts[LM_M_GRAM_LEVEL_MAX]) . . . 121
8.7.4.9 set_def_unk_word_prob(const prob_weight prob) . . . 122
8.8 uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType > Class Template Reference . . . 122
8.8.1 Detailed Description . . . 123
8.8.2 Member Typedef Documentation . . . 123
8.8.2.1 BASE . . . 123
8.8.2.2 TCtxIdProbEntry . . . 123
8.8.2.3 TWordIdPBEntry . . . 123
8.8.3 Constructor & Destructor Documentation . . . 123
8.8.3.1 c2w_array_trie(WordIndexType &p_word_index) . . . 123
8.8.3.2 ~c2w_array_trie() . . . 124
8.8.4 Member Function Documentation . . . 124
8.8.4.1 add_m_gram(const model_m_gram &gram) . . . 124
8.8.4.2 get_ctx_id(const phrase_length level_idx, const TShortId word_id, TLongId &ctx_id) const . . . 124
8.8.4.3 get_m_gram_payload(m_gram_query &query, MGramStatusEnum &status) const . . . 124
8.8.4.4 get_n_gram_payload(m_gram_query &query, MGramStatusEnum &status) const 124
8.8.4.5 get_unigram_payload(m_gram_query &query) const . . . 125
8.8.4.6 get_unk_word_prob() const . . . 125
8.8.4.7 is_post_grams() const . . . 125
8.8.4.8 log_model_type_info() const . . . 125
8.8.4.9 post_grams() . . . 125
8.8.4.10 post_m_grams() . . . 126
8.8.4.11 post_n_grams() . . . 126
8.8.4.12 pre_allocate(const size_t counts[LM_M_GRAM_LEVEL_MAX]) . . . 126
8.8.4.13 set_def_unk_word_prob(const prob_weight prob) . . . 126
8.9 uva::utils::containers::circular_queue< elem_type, capacity > Class Template Reference . . . 126
8.9.1 Detailed Description . . . 127
8.9.2 Constructor & Destructor Documentation . . . 127
8.9.2.1 circular_queue() . . . 127
8.9.2.2 circular_queue(const size_t num_elems, const elem_type *elems) . . . 127
8.9.2.3 circular_queue(const circular_queue &other, const size_t num_elems, const elem_type *elems) . . . 127
8.9.2.4 ~circular_queue() . . . 127
8.9.3 Member Function Documentation . . . 127
8.9.3.1 empty_queue() . . . 128
8.9.3.2 get_capacity() const . . . 128
8.9.3.3 get_elems() const . . . 128
8.9.3.4 get_size() const . . . 128

8.9.3.5 is_equal_last(const circular_queue &other, const size_t num_elems) const . . . 128
8.9.3.6 push_back(const elem_type &elem) . . . 129
8.9.3.7 push_back(const size_t num_elems, const elem_type *elems) . . . 130
8.9.3.8 tail_to_string(const size_t num_elems) const . . . 130
8.10 uva::smt::bpbd::client::client_config Struct Reference . . . 130
8.10.1 Detailed Description . . . 130
8.10.2 Member Data Documentation . . . 131
8.10.2.1 is_pre_process . . . 131
8.10.2.2 m_max_sent . . . 131
8.10.2.3 m_min_sent . . . 131
8.10.2.4 m_port . . . 131
8.10.2.5 m_server . . . 131
8.10.2.6 m_source_file . . . 131
8.10.2.7 m_source_lang . . . 131
8.10.2.8 m_target_file . . . 131
8.10.2.9 m_target_lang . . . 131
8.11 uva::smt::bpbd::server::lm::dictionary::counting_word_index Class Reference . . . 131
8.11.1 Detailed Description . . . 132
8.11.2 Constructor & Destructor Documentation . . . 132
8.11.2.1 counting_word_index(const float mem_factor) . . . 132
8.11.3 Member Function Documentation . . . 133
8.11.3.1 count_word(const text_piece_reader &word, prob_weight prob) . . . 133
8.11.3.2 do_post_actions() . . . 133
8.11.3.3 do_post_word_count() . . . 133
8.11.3.4 is_post_actions_needed() const . . . 133
8.11.3.5 is_word_counts_needed() const . . . 133
8.11.3.6 is_word_index_continuous() . . . 134
8.11.3.7 is_word_registering_needed() const . . . 134
8.11.3.8 register_word(const text_piece_reader &token) . . . 134
8.12 uva::utils::file::cstyle_file_reader Class Reference . . . 134
8.12.1 Detailed Description . . . 135
8.12.2 Constructor & Destructor Documentation . . . 135
8.12.2.1 cstyle_file_reader(const char *fileName) . . . 135
8.12.2.2 cstyle_file_reader(const string &file_name) . . . 135
8.12.2.3 ~cstyle_file_reader() . . . 135
8.12.3 Member Function Documentation . . . 135
8.12.3.1 close() . . . 135
8.12.3.2 get_first_line(text_piece_reader &out) . . . 136
8.12.3.3 is_open() const . . . 136
8.12.3.4 log_reader_type_info() . . . 136

8.12.3.5 operator bool() const . . . 136
8.12.3.6 reset() . . . 136
8.13 uva::smt::bpbd::server::decoder::de_configurator Class Reference . . . 136
8.13.1 Detailed Description . . . 137
8.13.2 Member Function Documentation . . . 137
8.13.2.1 allocate_decoder(acr_bool_flag is_stop, const string &source_sent, string &target_sent) . . . 137
8.13.2.2 connect(const de_parameters &params) . . . 137
8.13.2.3 disconnect() . . . 137
8.13.2.4 dispose_decoder(sentence_decoder &dec) . . . 137
8.14 uva::smt::bpbd::server::decoder::de_parameters_struct Struct Reference . . . 138
8.14.1 Detailed Description . . . 138
8.14.2 Constructor & Destructor Documentation . . . 138
8.14.2.1 de_parameters_struct() . . . 138
8.14.2.2 de_parameters_struct(const de_parameters_struct &other) . . . 138
8.14.3 Member Function Documentation . . . 139
8.14.3.1 finalize() . . . 139
8.14.3.2 operator=(const de_parameters_struct &other) . . . 139
8.14.4 Member Data Documentation . . . 139
8.14.4.1 m_distortion . . . 139
8.14.4.2 m_ext_dist_left . . . 139
8.14.4.3 m_is_dist . . . 139
8.14.4.4 m_is_recombine . . . 139
8.14.4.5 m_max_s_phrase_len . . . 139
8.14.4.6 m_max_t_phrase_len . . . 139
8.14.4.7 m_num_best_trans . . . 140
8.14.4.8 m_phrase_penalty . . . 140
8.14.4.9 m_pruning_threshold . . . 140
8.14.4.10 m_stack_capacity . . . 140
8.14.4.11 m_word_penalty . . . 140
8.15 uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR > Class Template Reference . . . 140
8.15.1 Detailed Description . . . 141
8.15.2 Member Typedef Documentation . . . 141
8.15.2.1 ELEMENT_TYPE_PTR . . . 141
8.15.2.2 TElemType . . . 141
8.15.2.3 TIndexType . . . 141
8.15.3 Constructor & Destructor Documentation . . . 142
8.15.3.1 dynamic_stack_array() . . . 142
8.15.3.2 ~dynamic_stack_array() . . . 142

8.15.4 Member Function Documentation . . . 142
8.15.4.1 allocate() . . . 142
8.15.4.2 data() const . . . 142
8.15.4.3 has_data() const . . . 142
8.15.4.4 operator[](IDX_DATA_TYPE idx) const . . . 143
8.15.4.5 pre_allocate(const IDX_DATA_TYPE capacity) . . . 143
8.15.4.6 shrink() . . . 143
8.15.4.7 size() const . . . 143
8.15.4.8 sort() . . . 144
8.15.4.9 sort(typename T_IS_COMPARE_FUNC< ELEMENT_TYPE >::func_type is_less_func) . . . 144
8.15.5 Member Data Documentation . . . 144
8.15.5.1 MAX_SIZE_TYPE_VALUE . . . 144
8.15.5.2 PARAMETERS_SIZE_BYTES . . . 144
8.16 uva::utils::containers::ELEMENT_DEALLOC_FUNC< ELEM_TYPE > Struct Template Reference . . . 144
8.16.1 Detailed Description . . . 145
8.16.2 Member Typedef Documentation . . . 145
8.16.2.1 func_ptr . . . 145
8.16.2.2 func_type . . . 145
8.16.3 Member Data Documentation . . . 145
8.16.3.1 NULL_FUNC_PTR . . . 145
8.17 uva::utils::file::file_stream_reader Class Reference . . . 145
8.17.1 Detailed Description . . . 146
8.17.2 Constructor & Destructor Documentation . . . 146
8.17.2.1 file_stream_reader(const char *fileName) . . . 146
8.17.2.2 ~file_stream_reader() . . . 146
8.17.3 Member Function Documentation . . . 146
8.17.3.1 close() . . . 146
8.17.3.2 get_first_line(text_piece_reader &out) . . . 147
8.17.3.3 is_open() const . . . 147
8.17.3.4 log_reader_type_info() . . . 147
8.17.3.5 operator bool() const . . . 147
8.17.3.6 reset() . . . 147
8.18 uva::utils::containers::fixed_size_hashmap< ELEMENT_TYPE, KEY_TYPE, IDX_TYPE > Class Template Reference . . . 147
8.18.1 Detailed Description . . . 148
8.18.2 Member Typedef Documentation . . . 148
8.18.2.1 TElemType . . . 148
8.18.3 Constructor & Destructor Documentation . . . 148
8.18.3.1 fixed_size_hashmap(const double buckets_factor, const IDX_TYPE num_elems) 148

8.18.3.2 ~fixed_size_hashmap() . . . 150
8.18.4 Member Function Documentation . . . 150
8.18.4.1 add_new_element(const uint_fast64_t key_uid) . . . 150
8.18.4.2 get_element(const uint_fast64_t key_uid, const KEY_TYPE &key) const . . . 150
8.18.5 Member Data Documentation . . . 150
8.18.5.1 MAX_ELEMENT_INDEX . . . 151
8.18.5.2 MIN_ELEMENT_INDEX . . . 151
8.18.5.3 NO_ELEMENT_INDEX . . . 151
8.19 uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType > Class Template Reference . . . 151
8.19.1 Detailed Description . . . 152
8.19.2 Member Typedef Documentation . . . 152
8.19.2.1 BASE . . . 152
8.19.2.2 T_M_Gram_PB_Entry . . . 152
8.19.2.3 T_M_Gram_Prob_Entry . . . 152
8.19.3 Constructor & Destructor Documentation . . . 152
8.19.3.1 g2d_map_trie(WordIndexType &word_index) . . . 152
8.19.3.2 ~g2d_map_trie() . . . 152
8.19.4 Member Function Documentation . . . 153
8.19.4.1 add_m_gram(const model_m_gram &gram) . . . 153
8.19.4.2 get_m_gram_payload(m_gram_query &query, MGramStatusEnum &status) const . . . 153
8.19.4.3 get_n_gram_payload(m_gram_query &query, MGramStatusEnum &status) const 153
8.19.4.4 get_unigram_payload(m_gram_query &query) const . . . 153
8.19.4.5 get_unk_word_prob() const . . . 154
8.19.4.6 log_model_type_info() const . . . 154
8.19.4.7 pre_allocate(const size_t counts[LM_M_GRAM_LEVEL_MAX]) . . . 154
8.19.4.8 set_def_unk_word_prob(const prob_weight prob) . . . 154
8.20 uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CAC↩ HE_BUCKETS_FACTOR > Class Template Reference . . . 154
8.20.1 Detailed Description . . . 155
8.20.2 Member Typedef Documentation . . . 155
8.20.2.1 BASE . . . 155
8.20.3 Constructor & Destructor Documentation . . . 156
8.20.3.1 generic_trie_base(WordIndexType &word_index) . . . 156
8.20.3.2 ~generic_trie_base() . . . 156
8.20.4 Member Function Documentation . . . 156
8.20.4.1 add_m_gram(const model_m_gram &gram) . . . 156
8.20.4.2 execute(m_gram_query &query) const . . . 156
8.20.4.3 get_m_gram_payload(m_gram_query &query, MGramStatusEnum &status) const . . . 157
8.20.4.4 get_n_gram_payload(m_gram_query &query, MGramStatusEnum &status) const 158

8.20.4.5 get_unigram_payload(m_gram_query &query, MGramStatusEnum &status) const 158
8.20.4.6 get_unk_word_prob() const . . . 158
8.20.4.7 is_context_needed() . . . 158
8.20.4.8 is_m_gram_potentially_present(m_gram_query &query, MGramStatusEnum &status) const . . . 159
8.20.4.9 log_model_type_info() const . . . 160
8.20.4.10 pre_allocate(const size_t counts[LM_M_GRAM_LEVEL_MAX]) . . . 160
8.20.4.11 register_m_gram_cache(const model_m_gram &gram) . . . 160
8.20.5 Member Data Documentation . . . 160
8.20.5.1 FIRST_VALID_CTX_ID . . . 160
8.20.5.2 MGRAM_IDX_OFFSET . . . 160
8.20.5.3 N_GRAM_IDX_IN_M_N_ARR . . . 161
8.20.5.4 NEEDS_BITMAP_HASH_CACHE . . . 161
8.20.5.5 NUM_M_GRAM_LEVELS . . . 161
8.20.5.6 NUM_M_N_GRAM_LEVELS . . . 161
8.20.5.7 UNDEFINED_ARR_IDX . . . 161
8.21 uva::utils::containers::alloc::greedy_memory_allocator< T > Class Template Reference . . . 161
8.21.1 Detailed Description . . . 162
8.21.2 Member Typedef Documentation . . . 162
8.21.2.1 const_pointer . . . 162
8.21.2.2 const_reference . . . 162
8.21.2.3 difference_type . . . 162
8.21.2.4 pointer . . . 163
8.21.2.5 reference . . . 163
8.21.2.6 size_type . . . 163
8.21.2.7 value_type . . . 163
8.21.3 Constructor & Destructor Documentation . . . 163
8.21.3.1 greedy_memory_allocator(size_type numElems) . . . 163
8.21.3.2 greedy_memory_allocator(const greedy_memory_allocator &other) . . . 163
8.21.3.3 greedy_memory_allocator(const greedy_memory_allocator< U > &other) . . . 163
8.21.3.4 ~greedy_memory_allocator() . . . 163
8.21.4 Member Function Documentation . . . 164
8.21.4.1 address(reference obj) const . . . 164
8.21.4.2 address(const_reference obj) const . . . 164
8.21.4.3 allocate(size_type num, const_pointer cp=0) . . . 164
8.21.4.4 available() const . . . 164
8.21.4.5 construct(pointer ptr, const value_type &value) . . . 165
8.21.4.6 deallocate(pointer ptr, size_type num) . . . 166
8.21.4.7 destroy(pointer ptr) . . . 166
8.21.4.8 getStorageRef() const . . . 166

8.21.4.9 max_size() const . . . 166
8.21.5 Member Data Documentation . . . 166
8.21.5.1 _manager . . . 166
8.22 uva::utils::containers::greedy_memory_storage Class Reference . . . 167
8.22.1 Detailed Description . . . 167
8.22.2 Member Typedef Documentation . . . 167
8.22.2.1 size_type . . . 167
8.22.2.2 TStorageData . . . 167
8.22.3 Constructor & Destructor Documentation . . . 168
8.22.3.1 greedy_memory_storage() . . . 168
8.22.3.2 greedy_memory_storage(size_type numBytes) . . . 168
8.22.3.3 greedy_memory_storage(const greedy_memory_storage &source) . . . 168
8.22.3.4 ∼greedy_memory_storage() . . . 168
8.22.4 Member Function Documentation . . . 168
8.22.4.1 allocate(size_type num) . . . 168
8.22.4.2 getAvailableBytes() const . . . 168
8.22.4.3 getBufferSizeBytes() const . . . 169
8.22.5 Member Data Documentation . . . 169
8.22.5.1 _allocBytes . . . 169
8.22.5.2 _memoryBuffers . . . 169
8.22.5.3 _numBytes . . . 169
8.22.5.4 _pBuffer . . . 169
8.23 uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType > Class Template Reference . . . 169
8.23.1 Detailed Description . . . 170
8.23.2 Member Typedef Documentation . . . 170
8.23.2.1 BASE . . . 170
8.23.2.2 T_M_Gram_PB_Entry . . . 170
8.23.2.3 T_M_Gram_Prob_Entry . . . 170
8.23.3 Constructor & Destructor Documentation . . . 170
8.23.3.1 h2d_map_trie(WordIndexType &word_index) . . . 170
8.23.3.2 ∼h2d_map_trie() . . . 171
8.23.4 Member Function Documentation . . . 171
8.23.4.1 add_m_gram(const model_m_gram &gram) . . . 171
8.23.4.2 get_m_gram_payload(m_gram_query &query, MGramStatusEnum &status) const . . . 171
8.23.4.3 get_n_gram_payload(m_gram_query &query, MGramStatusEnum &status) const 171
8.23.4.4 get_unigram_payload(m_gram_query &query) const . . . 172
8.23.4.5 get_unk_word_prob() const . . . 172
8.23.4.6 log_model_type_info() const . . . 172
8.23.4.7 pre_allocate(const size_t counts[LM_M_GRAM_LEVEL_MAX]) . . . 172

8.23.4.8 set_def_unk_word_prob(const prob_weight prob) . . . 172
8.24 uva::smt::bpbd::server::lm::dictionary::hashing_word_index Class Reference . . . 173
8.24.1 Detailed Description . . . 173
8.24.2 Constructor & Destructor Documentation . . . 173
8.24.2.1 hashing_word_index(const float memory_factor) . . . 173
8.24.2.2 ∼hashing_word_index() . . . 173
8.24.3 Member Function Documentation . . . 174
8.24.3.1 get_number_of_words(const size_t num_words) const . . . 174
8.24.3.2 get_word_id(const text_piece_reader &token) const . . . 174
8.24.3.3 is_post_actions_needed() const . . . 174
8.24.3.4 is_word_counts_needed() const . . . 174
8.24.3.5 is_word_index_continuous() . . . 174
8.24.3.6 is_word_registering_needed() const . . . 174
8.24.3.7 register_word(const text_piece_reader &token) . . . 175
8.24.3.8 reserve(const size_t num_words) . . . 175
8.25 uva::smt::bpbd::common::messaging::id_manager< id_type > Class Template Reference . . . 175
8.25.1 Detailed Description . . . 175
8.25.2 Member Typedef Documentation . . . 176
8.25.2.1 scoped_lock . . . 176
8.25.3 Constructor & Destructor Documentation . . . 176
8.25.3.1 id_manager(const id_type min_id) . . . 176
8.25.4 Member Function Documentation . . . 176
8.25.4.1 get_min_id() const . . . 176
8.25.4.2 get_next_id() . . . 176
8.26 uva::smt::bpbd::server::lm::layered_trie_base< TrieType, WordIndexType, BITMAP_HASH_CAC↩ HE_BUCKETS_FACTOR > Class Template Reference . . . 176
8.26.1 Detailed Description . . . 177
8.26.2 Member Typedef Documentation . . . 177
8.26.2.1 BASE . . . 177
8.26.3 Constructor & Destructor Documentation . . . 178
8.26.3.1 layered_trie_base(WordIndexType &word_index) . . . 178
8.26.4 Member Function Documentation . . . 179
8.26.4.1 ensure_context(m_gram_query &query, MGramStatusEnum &status) const . . . 179
8.26.4.2 get_cached_context_id(const model_m_gram &gram, TLongId &result) const . . . 179
8.26.4.3 get_ctx_id(const phrase_length level_idx, const TShortId word_id, TLongId &ctx_id) const . . . 179
8.26.4.4 is_context_needed() . . . 180
8.26.4.5 pre_allocate(const size_t counts[LM_M_GRAM_LEVEL_MAX]) . . . 180
8.26.4.6 set_cache_context_id(const model_m_gram &gram, TLongId &ctx_id) . . . 180
8.27 uva::smt::bpbd::server::lm::arpa::lm_basic_builder< trie_type, reader_type > Class Template Reference . . . 180

8.27.1 Detailed Description . . . 181
8.27.2 Member Typedef Documentation . . . 181
8.27.2.1 WordIndexType . . . 181
8.27.3 Constructor & Destructor Documentation . . . 181
8.27.3.1 lm_basic_builder(const lm_parameters &params, trie_type &trie, reader_type &file) 181
8.27.3.2 ∼lm_basic_builder() . . . 181
8.27.4 Member Function Documentation . . . 181
8.27.4.1 build() . . . 181
8.28 uva::smt::bpbd::server::lm::lm_configurator Class Reference . . . 182
8.28.1 Detailed Description . . . 182
8.28.2 Member Function Documentation . . . 182
8.28.2.1 allocate_fast_query_proxy() . . . 182
8.28.2.2 allocate_slow_query_proxy() . . . 182
8.28.2.3 connect(const lm_parameters &params) . . . 182
8.28.2.4 disconnect() . . . 183
8.28.2.5 dispose_fast_query_proxy(lm_fast_query_proxy &query) . . . 183
8.28.2.6 dispose_slow_query_proxy(lm_slow_query_proxy &query) . . . 183
8.29 uva::smt::bpbd::server::lm::__executor::lm_exec_params Struct Reference . . . 183
8.29.1 Detailed Description . . . 183
8.29.2 Member Data Documentation . . . 184
8.29.2.1 m_lm_params . . . 184
8.29.2.2 m_query_file_name . . . 184
8.30 uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy Class Reference . . . 184
8.30.1 Detailed Description . . . 184
8.30.2 Constructor & Destructor Documentation . . . 184
8.30.2.1 ∼lm_fast_query_proxy() . . . 184
8.30.3 Member Function Documentation . . . 185
8.30.3.1 execute(const phrase_length num_words, const word_uid ∗word_ids)=0 . . . 185
8.30.3.2 execute(const phrase_length num_words, const word_uid ∗word_ids, phrase_↩ length &min_level)=0 . . . 185
8.30.3.3 get_begin_tag_uid() const =0 . . . 185
8.30.3.4 get_end_tag_uid() const =0 . . . 186
8.30.3.5 get_unk_word_prob() const =0 . . . 186
8.30.3.6 get_word_ids(text_piece_reader phrase, phrase_length &num_words, word_uid word_ids[tm::TM_MAX_TARGET_PHRASE_LEN]) const =0 . . . 186
8.31 uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type > Class Template Reference 186
8.31.1 Detailed Description . . . 187
8.31.2 Member Typedef Documentation . . . 187
8.31.2.1 word_index_type . . . 187
8.31.3 Constructor & Destructor Documentation . . . 187

8.31.3.1 lm_fast_query_proxy_local(const trie_type &trie, const prob_weight &unk_word_prob, const word_uid &begin_tag_uid, const word_uid &end_tag_uid) 187
8.31.3.2 ∼lm_fast_query_proxy_local() 188
8.31.4 Member Function Documentation 188
8.31.4.1 execute(const phrase_length num_words, const word_uid ∗word_ids) 188
8.31.4.2 execute(const phrase_length num_words, const word_uid ∗word_ids, phrase_length &min_level) 188
8.31.4.3 get_begin_tag_uid() const 188
8.31.4.4 get_end_tag_uid() const 188
8.31.4.5 get_m_gram_str(const phrase_length begin_word_idx, const phrase_length end_word_idx) const 189
8.31.4.6 get_query_str() const 189
8.31.4.7 get_report_interm_results(const phrase_length begin_word_idx, const phrase_length first_end_word_idx, const phrase_length last_end_word_idx) 189
8.31.4.8 get_unk_word_prob() const 189
8.31.4.9 get_word_ids(text_piece_reader phrase, phrase_length &num_words, word_uid word_ids[tm::TM_MAX_TARGET_PHRASE_LEN]) const 190
8.31.4.10 report_final_result() 190
8.32 uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight > Class Template Reference 190
8.32.1 Detailed Description 191
8.32.2 Constructor & Destructor Documentation 191
8.32.2.1 lm_gram_builder(const lm_parameters &params, WordIndexType &word_index, typename TAddGramFunct< WordIndexType >::func addGarmFunc) 191
8.32.2.2 ∼lm_gram_builder() 191
8.32.2.3 lm_gram_builder(const lm_gram_builder &orig) 191
8.32.3 Member Function Documentation 192
8.32.3.1 parse_line(text_piece_reader &data) 192
8.32.3.2 parse_to_gram(text_piece_reader &line) 192
8.32.3.3 unigram_to_prob(text_piece_reader &text, text_piece_reader &word, prob_weight &prob) 192
8.32.4 Member Data Documentation 193
8.32.4.1 m_add_garm_func 193
8.32.4.2 m_m_gram 193
8.32.4.3 m_params 193
8.32.4.4 m_token 193
8.32.4.5 m_word_idx 193
8.32.4.6 MAX_NUM_TOKENS_NGRAM_STR 193
8.32.4.7 MIN_NUM_TOKENS_NGRAM_STR 193
8.33 uva::smt::bpbd::server::lm::arpa::lm_gram_builder_factory< TrieType > Class Template Reference 194
8.33.1 Detailed Description 194
8.33.2 Member Typedef Documentation 194

8.33.2.1 WordIndexType . . . 194
8.33.3 Constructor & Destructor Documentation . . . 194
8.33.3.1 ∼lm_gram_builder_factory() . . . 194
8.33.4 Member Function Documentation . . . 194
8.33.4.1 get_builder(const lm_parameters &params, TrieType &trie, lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight > ∗∗ppBuilder) . . . 195
8.34 uva::smt::bpbd::server::lm::lm_parameters Struct Reference . . . 195
8.34.1 Detailed Description . . . 195
8.34.2 Member Function Documentation . . . 195
8.34.2.1 finalize() . . . 195
8.34.2.2 get_lm_weight() const . . . 196
8.34.2.3 is_lm_weight() const . . . 196
8.34.3 Member Data Documentation . . . 196
8.34.3.1 m_conn_string . . . 196
8.34.3.2 m_lambdas . . . 196
8.34.3.3 m_num_lambdas . . . 196
8.35 uva::smt::bpbd::server::lm::proxy::lm_proxy Class Reference . . . 196
8.35.1 Detailed Description . . . 197
8.35.2 Constructor & Destructor Documentation . . . 197
8.35.2.1 ∼lm_proxy() . . . 197
8.35.3 Member Function Documentation . . . 197
8.35.3.1 allocate_fast_query_proxy()=0 . . . 197
8.35.3.2 allocate_slow_query_proxy()=0 . . . 197
8.35.3.3 connect(const lm_parameters &params)=0 . . . 197
8.35.3.4 disconnect()=0 . . . 198
8.35.3.5 dispose_fast_query_proxy(lm_fast_query_proxy &query)=0 . . . 198
8.35.3.6 dispose_slow_query_proxy(lm_slow_query_proxy &query)=0 . . . 198
8.36 uva::smt::bpbd::server::lm::proxy::lm_proxy_local Class Reference . . . 198
8.36.1 Detailed Description . . . 199
8.36.2 Constructor & Destructor Documentation . . . 199
8.36.2.1 lm_proxy_local() . . . 199
8.36.2.2 ∼lm_proxy_local() . . . 199
8.36.3 Member Function Documentation . . . 199
8.36.3.1 allocate_fast_query_proxy() . . . 199
8.36.3.2 allocate_slow_query_proxy() . . . 200
8.36.3.3 connect(const lm_parameters &params) . . . 200
8.36.3.4 disconnect() . . . 200
8.36.3.5 dispose_fast_query_proxy(lm_fast_query_proxy &query) . . . 200
8.36.3.6 dispose_slow_query_proxy(lm_slow_query_proxy &query) . . . 200
8.36.4 Member Data Documentation . . . 201

8.36.4.1 m_begin_tag_uid . . . 201
8.36.4.2 m_end_tag_uid . . . 201
8.36.4.3 m_model . . . 201
8.36.4.4 m_unk_word_prob . . . 201
8.36.4.5 m_word_index . . . 201
8.37 uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy Class Reference . . . 201
8.37.1 Detailed Description . . . 201
8.37.2 Constructor & Destructor Documentation . . . 202
8.37.2.1 ∼lm_slow_query_proxy() . . . 202
8.37.3 Member Function Documentation . . . 202
8.37.3.1 execute(text_piece_reader &line)=0 . . . 202
8.38 uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local< trie_type > Class Template Reference . . . 202
8.38.1 Detailed Description . . . 203
8.38.2 Member Typedef Documentation . . . 203
8.38.2.1 word_index_type . . . 203
8.38.3 Constructor & Destructor Documentation . . . 203
8.38.3.1 lm_slow_query_proxy_local(const trie_type &trie) . . . 203
8.38.3.2 ∼lm_slow_query_proxy_local() . . . 203
8.38.4 Member Function Documentation . . . 203
8.38.4.1 execute(text_piece_reader &line) . . . 203
8.38.4.2 get_m_gram_str(const phrase_length begin_word_idx, const phrase_length end_word_idx) const . . . 205
8.38.4.3 get_query_str() const . . . 205
8.38.4.4 get_report_interm_results(const phrase_length begin_word_idx, const phrase↩ _length first_end_word_idx, const phrase_length last_end_word_idx) . . . 205
8.38.4.5 report_final_result() . . . 205
8.38.4.6 set_tokens_and_word_ids(text_piece_reader phrase) . . . 206
8.39 uva::utils::logging::logger Class Reference . . . 206
8.39.1 Detailed Description . . . 206
8.39.2 Constructor & Destructor Documentation . . . 206
8.39.2.1 ∼logger() . . . 206
8.39.3 Member Function Documentation . . . 206
8.39.3.1 get(debug_levels_enum level) . . . 206
8.39.3.2 get(debug_levels_enum level, const char ∗file, const char ∗func, const char ∗line) 207
8.39.3.3 get_curr_level_str() . . . 207
8.39.3.4 get_reporting_level() . . . 207
8.39.3.5 get_reporting_levels(vector< string > ∗p_reporting_levels) . . . 207
8.39.3.6 is_progress_bar_on() . . . 208
8.39.3.7 is_relevant_level(const debug_levels_enum &level) . . . 208
8.39.3.8 set_reporting_level(const string level) . . . 208

8.39.3.9 start_progress_bar(const string &msg) . . . 208
8.39.3.10 stop_progress_bar() . . . 208
8.39.3.11 update_progress_bar() . . . 208
8.40 uva::utils::logging::logging_synch Struct Reference . . . 209
8.40.1 Detailed Description . . . 209
8.40.2 Member Typedef Documentation . . . 209
8.40.2.1 rec_scoped_lock . . . 209
8.40.3 Member Data Documentation . . . 209
8.40.3.1 mv . . . 209
8.41 uva::smt::bpbd::server::lm::m_grams::m_gram_payload_s Struct Reference . . . 209
8.41.1 Detailed Description . . . 210
8.41.2 Constructor & Destructor Documentation . . . 210
8.41.2.1 m_gram_payload_s() . . . 210
8.41.2.2 m_gram_payload_s(prob_weight prob, prob_weight back) . . . 210
8.41.3 Member Data Documentation . . . 210
8.41.3.1 m_back . . . 210
8.41.3.2 m_prob . . . 210
8.42 uva::smt::bpbd::server::lm::m_gram_query Class Reference . . . 210
8.42.1 Detailed Description . . . 211
8.42.2 Member Typedef Documentation . . . 211
8.42.2.1 payload_ptr . . . 211
8.42.3 Constructor & Destructor Documentation . . . 212
8.42.3.1 m_gram_query() . . . 212
8.42.4 Member Function Documentation . . . 213
8.42.4.1 get_curr_begin_word_id() const . . . 213
8.42.4.2 get_curr_ctx_ref() . . . 213
8.42.4.3 get_curr_end_word_id() const . . . 213
8.42.4.4 get_curr_level() const . . . 213
8.42.4.5 get_curr_level_m1() const . . . 213
8.42.4.6 get_curr_level_m2() const . . . 214
8.42.4.7 get_curr_m_gram_hash() . . . 214
8.42.4.8 get_curr_m_gram_id(uint8_t &len_bytes) . . . 214
8.42.4.9 get_curr_payload_ref() . . . 214
8.42.4.10 get_curr_uni_gram_word_id() const . . . 214
8.42.4.11 get_query_begin_word_idx() const . . . 215
8.42.4.12 get_query_end_word_idx() const . . . 215
8.42.4.13 is_curr_uni_gram() const . . . 215
8.42.4.14 is_not_finished() const . . . 215
8.42.4.15 operator[](const phrase_length idx) const . . . 215
8.42.4.16 set_curr_payload(const void *payload) . . . 215

8.42.4.17 set_data(const phrase_length num_words, const word_uid *word_ids) 216
8.42.4.18 set_word_indxes(const phrase_length sub_query_begin_word_idx, const phrase_length sub_sub_query_first_end_word_idx, const phrase_length sub_query_end_word_idx) 216
8.42.4.19 set_word_indxes(const phrase_length sub_query_begin_word_idx, const phrase_length sub_query_end_word_idx) 216
8.42.5 Friends And Related Function Documentation 216
8.42.5.1 operator<< 216
8.42.6 Member Data Documentation 217
8.42.6.1 m_curr_begin_word_idx 217
8.42.6.2 m_curr_end_word_idx 217
8.42.6.3 m_probs 217
8.43 uva::utils::containers::mem_increase_strategy Class Reference 217
8.43.1 Detailed Description 217
8.43.2 Constructor & Destructor Documentation 218
8.43.2.1 mem_increase_strategy(const mem_inc_types_enum &stype, const TCapacityIncFunct get_capacity_inc_func, const size_t min_mem_inc, const size_t mem_inc_factor) 218
8.43.2.2 mem_increase_strategy() 219
8.43.2.3 mem_increase_strategy(const mem_increase_strategy &other) 219
8.43.3 Member Function Documentation 219
8.43.3.1 get_new_capacity(const size_t capacity) const 219
8.43.3.2 get_strategy_info() const 219
8.44 uva::utils::file::memory_mapped_file_reader Class Reference 219
8.44.1 Detailed Description 220
8.44.2 Constructor & Destructor Documentation 221
8.44.2.1 memory_mapped_file_reader(const char *fileName) 221
8.44.3 Member Function Documentation 221
8.44.3.1 close() 221
8.44.3.2 get_first_line(text_piece_reader &out) 221
8.44.3.3 is_open() const 221
8.44.3.4 log_reader_type_info() 221
8.44.3.5 operator bool() const 221
8.45 uva::utils::monitor::memory_usage Struct Reference 222
8.45.1 Detailed Description 222
8.45.2 Constructor & Destructor Documentation 222
8.45.2.1 memory_usage() 222
8.45.3 Member Data Documentation 222
8.45.3.1 vmhwm 222
8.45.3.2 vmpeak 222
8.45.3.3 vmrss 223

8.45.3.4 vmsize . . . 223
8.46 uva::smt::bpbd::server::lm::m_grams::model_m_gram Class Reference . . . 223
8.46.1 Detailed Description . . . 224
8.46.2 Member Typedef Documentation . . . 224
8.46.2.1 BASE . . . 224
8.46.3 Constructor & Destructor Documentation . . . 224
8.46.3.1 model_m_gram(phrase_length actual_level) . . . 224
8.46.4 Member Function Documentation . . . 224
8.46.4.1 get_hash() const . . . 224
8.46.4.2 get_next_new_token() . . . 224
8.46.4.3 is_unk_unigram() const . . . 224
8.46.4.4 prepare_for_adding(WordIndexType &word_index) . . . 225
8.46.4.5 start_new_m_gram() . . . 225
8.46.5 Friends And Related Function Documentation . . . 225
8.46.5.1 operator<< . . . 225
8.46.6 Member Data Documentation . . . 225
8.46.6.1 m_back_off . . . 225
8.46.6.2 m_payload . . . 225
8.46.6.3 m_prob . . . 226
8.47 uva::smt::bpbd::server::decoder::stack::multi_stack Class Reference . . . 226
8.47.1 Detailed Description . . . 226
8.47.2 Constructor & Destructor Documentation . . . 226
8.47.2.1 multi_stack(const de_parameters &params, acr_bool_flag is_stop, const string &source_sent, const sentence_data_map &sent_data, const rm_query_proxy &rm_query, lm_fast_query_proxy &lm_query) . . . 226
8.47.2.2 ∼multi_stack() . . . 226
8.47.3 Member Function Documentation . . . 227
8.47.3.1 add_stack_state(stack_state_ptr new_state) . . . 227
8.47.3.2 expand() . . . 227
8.47.3.3 get_best_trans(string &target_sent) const . . . 227
8.48 uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type > Class Template Reference . . . 227
8.48.1 Detailed Description . . . 228
8.48.2 Constructor & Destructor Documentation . . . 228
8.48.2.1 optimizing_word_index(const float memory_factor) . . . 228
8.48.2.2 ∼optimizing_word_index() . . . 228
8.48.3 Member Function Documentation . . . 228
8.48.3.1 count_word(const text_piece_reader &word, prob_weight prob) . . . 228
8.48.3.2 do_post_actions() . . . 229
8.48.3.3 do_post_word_count() . . . 229
8.48.3.4 get_number_of_words(const size_t num_words) const . . . 229

8.48.3.5 get_word_id(const text_piece_reader &token) const . . . 229
8.48.3.6 is_post_actions_needed() const . . . 230
8.48.3.7 is_word_counts_needed() const . . . 230
8.48.3.8 is_word_index_continuous() . . . 230
8.48.3.9 is_word_registering_needed() const . . . 230
8.48.3.10 register_word(const text_piece_reader &token) . . . 231
8.48.3.11 reserve(const size_t num_words) . . . 231
8.49 uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_LENGTH > Class Template Reference . . . 231
8.49.1 Detailed Description . . . 232
8.49.2 Member Typedef Documentation . . . 232
8.49.2.1 m_gram_id_type . . . 232
8.49.3 Constructor & Destructor Documentation . . . 232
8.49.3.1 phrase_base(word_uid *word_ids, phrase_length actual_level) . . . 232
8.49.3.2 phrase_base() . . . 232
8.49.4 Member Function Documentation . . . 232
8.49.4.1 create_phrase_id(const phrase_length begin_word_idx, const phrase_length number_of_words, TM_Gram_Id_Value_Ptr &p_m_gram_id) const . . . 232
8.49.4.2 get_first_word_idx() const . . . 234
8.49.4.3 get_last_word_id() const . . . 234
8.49.4.4 get_last_word_idx() const . . . 234
8.49.4.5 get_num_words() const . . . 234
8.49.4.6 get_phrase_id_ref(const phrase_length begin_word_idx, const phrase_length number_of_words, uint8_t &len_bytes) . . . 235
8.49.4.7 operator[](const phrase_length word_idx) const . . . 235
8.49.4.8 set_word_ids(const phrase_length num_words, const word_uid *word_ids) . . . 235
8.49.4.9 word_ids() const . . . 235
8.50 uva::smt::bpbd::server::decoder::sentence::phrase_data_entry Struct Reference . . . 236
8.50.1 Detailed Description . . . 236
8.50.2 Constructor & Destructor Documentation . . . 236
8.50.2.1 phrase_data_entry() . . . 236
8.50.2.2 ~phrase_data_entry() . . . 236
8.50.3 Member Data Documentation . . . 236
8.50.3.1 future_cost . . . 236
8.50.3.2 m_begin_ch_idx . . . 237
8.50.3.3 m_end_ch_idx . . . 237
8.50.3.4 m_phrase_uid . . . 237
8.50.3.5 m_source_entry . . . 237
8.51 uva::smt::bpbd::server::lm::m_grams::query_m_gram Class Reference . . . 237
8.51.1 Detailed Description . . . 237
8.51.2 Member Typedef Documentation . . . 238

8.51.2.1 BASE . . . 238
8.51.3 Constructor & Destructor Documentation . . . 238
8.51.3.1 query_m_gram() . . . 238
8.51.4 Member Function Documentation . . . 238
8.51.4.1 get_hash(phrase_length begin_word_idx, const phrase_length end_word_idx) const . . . 238
8.51.4.2 set_m_gram(const phrase_length num_words, const word_uid *word_ids) . . . 238
8.51.5 Friends And Related Function Documentation . . . 238
8.51.5.1 operator<< . . . 238
8.52 uva::utils::containers::alloc::greedy_memory_allocator< T >::rebind< U > Struct Template Reference 239
8.52.1 Detailed Description . . . 239
8.52.2 Member Typedef Documentation . . . 239
8.52.2.1 other . . . 239
8.53 uva::smt::bpbd::server::rm::builders::rm_basic_builder< model_type, reader_type > Class Template Reference . . . 239
8.53.1 Detailed Description . . . 240
8.53.2 Constructor & Destructor Documentation . . . 240
8.53.2.1 rm_basic_builder(const rm_parameters &params, model_type &model, reader_type &reader) . . . 240
8.53.3 Member Function Documentation . . . 240
8.53.3.1 build() . . . 240
8.53.3.2 count_source_target_phrases(tm_query_proxy &query) . . . 240
8.53.3.3 parse_rm_file(tm_query_proxy &query) . . . 241
8.53.3.4 process_entry_weights(text_piece_reader &rest, rm_entry &entry) . . . 241
8.53.3.5 process_source_entries(tm_query_proxy &query) . . . 241
8.54 uva::smt::bpbd::server::rm::models::rm_basic_model Class Reference . . . 241
8.54.1 Detailed Description . . . 242
8.54.2 Member Typedef Documentation . . . 242
8.54.2.1 rm_entry_map . . . 242
8.54.3 Constructor & Destructor Documentation . . . 242
8.54.3.1 rm_basic_model() . . . 242
8.54.3.2 ~rm_basic_model() . . . 242
8.54.4 Member Function Documentation . . . 243
8.54.4.1 add_entry(const phrase_uid &source_uid, const phrase_uid &target_uid) . . . 243
8.54.4.2 find_begin_end_entries() . . . 244
8.54.4.3 find_unk_entry() . . . 244
8.54.4.4 get_begin_tag_entry() const . . . 244
8.54.4.5 get_end_tag_entry() const . . . 244
8.54.4.6 get_entry(const phrase_uid uid) const . . . 244
8.54.4.7 get_entry(const phrase_uid &source_uid, const phrase_uid &target_uid) const . . . 245
8.54.4.8 is_num_entries_needed() const . . . 245

8.54.4.9 is_unk_entry(const rm_entry *entry) const . . . 245
8.54.4.10 log_model_type_info() . . . 245
8.54.4.11 set_num_entries(size_t num_entries) . . . 246
8.54.5 Member Data Documentation . . . 247
8.54.5.1 BEGIN_SENT_TAG_UID . . . 247
8.54.5.2 END_SENT_TAG_UID . . . 247
8.54.5.3 SOURCE_UNK_UID . . . 247
8.54.5.4 TARGET_UNK_UID . . . 247
8.55 uva::smt::bpbd::server::rm::rm_configurator Class Reference . . . 247
8.55.1 Detailed Description . . . 247
8.55.2 Member Function Documentation . . . 247
8.55.2.1 allocate_query_proxy() . . . 247
8.55.2.2 connect(const rm_parameters &params) . . . 248
8.55.2.3 disconnect() . . . 248
8.55.2.4 dispose_query_proxy(rm_query_proxy &query) . . . 248
8.56 uva::smt::bpbd::server::rm::models::rm_entry_temp< num_features > Class Template Reference . . . 248
8.56.1 Detailed Description . . . 249
8.56.2 Constructor & Destructor Documentation . . . 249
8.56.2.1 rm_entry_temp() . . . 249
8.56.2.2 ~rm_entry_temp() . . . 249
8.56.3 Member Function Documentation . . . 249
8.56.3.1 get_weight(const reordering_orientation orient) const . . . 249
8.56.3.2 get_weights() const . . . 250
8.56.3.3 operator==(const phrase_uid &uid) const . . . 250
8.56.3.4 operator==(const rm_entry_temp &other) const . . . 250
8.56.3.5 operator[](size_t idx) . . . 250
8.56.3.6 set_entry_uid(const phrase_uid &uid) . . . 250
8.56.4 Friends And Related Function Documentation . . . 251
8.56.4.1 operator<< . . . 251
8.56.5 Member Data Documentation . . . 251
8.56.5.1 NUM_FEATURES . . . 251
8.57 uva::smt::bpbd::server::rm::rm_parameters Struct Reference . . . 251
8.57.1 Detailed Description . . . 251
8.57.2 Member Function Documentation . . . 252
8.57.2.1 finalize() . . . 252
8.57.3 Member Data Documentation . . . 252
8.57.3.1 m_conn_string . . . 252
8.57.3.2 m_lambdas . . . 252
8.57.3.3 m_num_lambdas . . . 252
8.58 uva::smt::bpbd::server::rm::proxy::rm_proxy Class Reference . . . 252

8.58.1 Detailed Description . . . 252
8.58.2 Constructor & Destructor Documentation . . . 253
8.58.2.1 ∼rm_proxy() . . . 253
8.58.3 Member Function Documentation . . . 253
8.58.3.1 allocate_query_proxy()=0 . . . 253
8.58.3.2 connect(const rm_parameters &params)=0 . . . 253
8.58.3.3 disconnect()=0 . . . 253
8.58.3.4 dispose_query_proxy(rm_query_proxy &query)=0 . . . 253
8.59 uva::smt::bpbd::server::rm::proxy::rm_proxy_local Class Reference . . . 253
8.59.1 Detailed Description . . . 254
8.59.2 Constructor & Destructor Documentation . . . 254
8.59.2.1 rm_proxy_local() . . . 254
8.59.2.2 ∼rm_proxy_local() . . . 254
8.59.3 Member Function Documentation . . . 254
8.59.3.1 allocate_query_proxy() . . . 254
8.59.3.2 connect(const rm_parameters &params) . . . 255
8.59.3.3 disconnect() . . . 255
8.59.3.4 dispose_query_proxy(rm_query_proxy &query) . . . 255
8.59.3.5 load_model_data(char const ∗model_name, const rm_parameters &params) . . . 255
8.60 uva::smt::bpbd::server::rm::models::rm_query< model_type > Class Template Reference . . . 255
8.60.1 Detailed Description . . . 256
8.60.2 Member Typedef Documentation . . . 256
8.60.2.1 query_map . . . 256
8.60.3 Constructor & Destructor Documentation . . . 256
8.60.3.1 rm_query(const model_type &model) . . . 256
8.60.3.2 ∼rm_query() . . . 256
8.60.4 Member Function Documentation . . . 256
8.60.4.1 execute(const vector< phrase_uid > &st_ids) . . . 256
8.60.4.2 get_reordering(const phrase_uid uid) const . . . 257
8.61 uva::smt::bpbd::server::rm::proxy::rm_query_proxy Class Reference . . . 257
8.61.1 Detailed Description . . . 257
8.61.2 Constructor & Destructor Documentation . . . 257
8.61.2.1 ∼rm_query_proxy() . . . 258
8.61.3 Member Function Documentation . . . 258
8.61.3.1 execute(const vector< phrase_uid > &st_ids)=0 . . . 258
8.61.3.2 get_begin_tag_reordering() const =0 . . . 258
8.61.3.3 get_end_tag_reordering() const =0 . . . 258
8.61.3.4 get_reordering(const phrase_uid uid) const =0 . . . 258
8.62 uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type > Class Template Reference 259
8.62.1 Detailed Description . . . 259

8.62.2 Constructor & Destructor Documentation . . . 259
8.62.2.1 rm_query_proxy_local(const model_type &model, const rm_entry &begin_tag_↩ entry, const rm_entry &end_tag_entry) . . . 259
8.62.2.2 ∼rm_query_proxy_local() . . . 259
8.62.3 Member Function Documentation . . . 260
8.62.3.1 execute(const vector< phrase_uid > &st_ids) . . . 260
8.62.3.2 get_begin_tag_reordering() const . . . 260
8.62.3.3 get_end_tag_reordering() const . . . 260
8.62.3.4 get_reordering(const phrase_uid uid) const . . . 260
8.63 uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_GramData< TPayloadType > Struct Template Reference . . . 261
8.63.1 Detailed Description . . . 261
8.63.2 Member Typedef Documentation . . . 261
8.63.2.1 SELF . . . 261
8.63.2.2 TM_Gram_Id . . . 261
8.63.3 Constructor & Destructor Documentation . . . 261
8.63.3.1 S_M_GramData() . . . 261
8.63.3.2 ∼S_M_GramData() . . . 262
8.63.4 Member Function Documentation . . . 262
8.63.4.1 operator==(const TM_Gram_Id &id) const . . . 262
8.63.5 Member Data Documentation . . . 262
8.63.5.1 m_id . . . 262
8.63.5.2 m_payload . . . 262
8.64 uva::smt::bpbd::server::lm::__W2CArrayTrie::S_M_GramData< PAYLOAD_TYPE > Struct Template Reference . . . 262
8.64.1 Detailed Description . . . 263
8.64.2 Member Data Documentation . . . 263
8.64.2.1 id . . . 263
8.64.2.2 m_mem_strat . . . 263
8.64.2.3 payload . . . 263
8.65 uva::smt::bpbd::server::lm::__G2DMapTrie::S_M_GramData< TPayloadType, TWordIdType > Struct Template Reference . . . 263
8.65.1 Detailed Description . . . 264
8.65.2 Member Typedef Documentation . . . 264
8.65.2.1 SELF . . . 264
8.65.2.2 TM_Gram_Id . . . 264
8.65.3 Constructor & Destructor Documentation . . . 264
8.65.3.1 S_M_GramData() . . . 264
8.65.3.2 ∼S_M_GramData() . . . 265
8.65.4 Member Function Documentation . . . 265
8.65.4.1 operator==(const T_Gram_Id_Key &key) const . . . 265

8.65.5 Member Data Documentation . . . 265
8.65.5.1 m_id . . . 265
8.65.5.2 m_payload . . . 265
8.66 uva::smt::bpbd::server::decoder::sentence::sentence_decoder Class Reference . . . 265
8.66.1 Detailed Description . . . 266
8.66.2 Constructor & Destructor Documentation . . . 266
8.66.2.1 sentence_decoder(const de_parameters &params, acr_bool_flag is_stop, const string &source_sent, string &target_sent) . . . 266
8.66.2.2 ~sentence_decoder() . . . 266
8.66.3 Member Function Documentation . . . 266
8.66.3.1 compute_futue_costs() . . . 266
8.66.3.2 count_words(const string &sentence) . . . 266
8.66.3.3 initialize_future_costs(const size_t &start_idx, const size_t &end_idx) . . . 267
8.66.3.4 perform_translation() . . . 267
8.66.3.5 query_reordering_model() . . . 267
8.66.3.6 query_translation_model() . . . 267
8.66.3.7 translate() . . . 267
8.67 uva::smt::bpbd::server::server_parameters Struct Reference . . . 267
8.67.1 Detailed Description . . . 268
8.67.2 Member Function Documentation . . . 268
8.67.2.1 verify() . . . 268
8.67.3 Member Data Documentation . . . 268
8.67.3.1 m_de_params . . . 268
8.67.3.2 m_lm_params . . . 268
8.67.3.3 m_num_threads . . . 268
8.67.3.4 m_rm_params . . . 268
8.67.3.5 m_server_port . . . 268
8.67.3.6 m_source_lang . . . 268
8.67.3.7 m_target_lang . . . 269
8.67.3.8 m_tm_params . . . 269
8.68 uva::smt::bpbd::server::decoder::stack::stack_data Struct Reference . . . 269
8.68.1 Detailed Description . . . 269
8.68.2 Constructor & Destructor Documentation . . . 269
8.68.2.1 stack_data(const de_parameters &params, acr_bool_flag is_stop, const string &source_sent, const sentence_data_map &sent_data, const rm_query_proxy &rm_query, lm_fast_query_proxy &lm_query, const add_new_state_function &add_state) . . . 269
8.68.3 Member Data Documentation . . . 270
8.68.3.1 m_add_state . . . 270
8.68.3.2 m_is_stop . . . 270
8.68.3.3 m_lm_query . . . 270

8.68.3.4 m_params . . . 270
8.68.3.5 m_rm_query . . . 270
8.68.3.6 m_sent_data . . . 270
8.68.3.7 m_source_sent . . . 270
8.69 uva::smt::bpbd::server::decoder::stack::stack_level Class Reference . . . 270
8.69.1 Detailed Description . . . 271
8.69.2 Constructor & Destructor Documentation . . . 271
8.69.2.1 stack_level(const de_parameters &params, acr_bool_flag is_stop) . . . 271
8.69.2.2 ∼stack_level() . . . 271
8.69.3 Member Function Documentation . . . 271
8.69.3.1 add_before(stack_state_ptr curr_state, stack_state_ptr new_state) . . . 271
8.69.3.2 add_last(stack_state_ptr new_state) . . . 273
8.69.3.3 add_state(stack_state_ptr new_state) . . . 273
8.69.3.4 expand() . . . 273
8.69.3.5 find_recombine(stack_state_ptr &curr_state, stack_state &new_state) . . . 273
8.69.3.6 get_best_trans(string &target_sent) const . . . 273
8.69.3.7 get_size() const . . . 274
8.69.3.8 insert_as_first(stack_state_ptr state) . . . 274
8.69.3.9 insert_as_last(stack_state_ptr state) . . . 274
8.69.3.10 insert_before(stack_state_ptr curr_state, stack_state_ptr new_state) . . . 274
8.69.3.11 insert_between(stack_state_ptr prev, stack_state_ptr next, stack_state_ptr state) 274
8.69.3.12 is_space_left() const . . . 275
8.69.3.13 prune_states() . . . 275
8.69.3.14 remember_best_score() . . . 275
8.69.3.15 remove_from_level(stack_state_ptr state) . . . 275
8.70 uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH > Class Template Reference . . . 275
8.70.1 Detailed Description . . . 276
8.70.2 Member Typedef Documentation . . . 276
8.70.2.1 state_data . . . 276
8.70.3 Constructor & Destructor Documentation . . . 277
8.70.3.1 stack_state_templ(const stack_data &data) . . . 277
8.70.3.2 stack_state_templ(stack_state_ptr parent) . . . 277
8.70.3.3 stack_state_templ(stack_state_ptr parent, const int32_t begin_pos, const int32_t end_pos, const typename state_data::covered_info &covered, tm_const_target_entry *target) . . . 277
8.70.3.4 ∼stack_state_templ() . . . 277
8.70.4 Member Function Documentation . . . 277
8.70.4.1 count_and_prune(size_t state_count, stack_state_ptr tail) . . . 278
8.70.4.2 cut_the_tail(stack_state_ptr tail) . . . 279
8.70.4.3 expand() . . . 279

8.70.4.4 expand_left() . . . 279
8.70.4.5 expand_length(const size_t start_pos) . . . 279
8.70.4.6 expand_length_if_not_covered(int32_t &curr_pos, size_t &num_exp) . . . 279
8.70.4.7 expand_right() . . . 280
8.70.4.8 expand_trans(const size_t start_pos, const size_t end_pos) . . . 280
8.70.4.9 get_stack_level() const . . . 280
8.70.4.10 get_translation(string &target_sent) const . . . 280
8.70.4.11 is_above_threshold(const prob_weight &score_bound) const . . . 280
8.70.4.12 merge_recomb_from(const stack_state_ptr recomb_from, const size_t recomb↩ _from_count) . . . 281
8.70.4.13 operator"!=(const stack_state &other) const . . . 281
8.70.4.14 operator<(const stack_state &other) const . . . 281
8.70.4.15 operator==(const stack_state &other) const . . . 282
8.70.4.16 recombine_from(stack_state_ptr other_state) . . . 282
8.70.5 Friends And Related Function Documentation . . . 282
8.70.5.1 stack_level . . . 282
8.71 uva::utils::monitor::stat_monitor Class Reference . . . 283
8.71.1 Detailed Description . . . 283
8.71.2 Member Function Documentation . . . 283
8.71.2.1 get_cpu_time() . . . 283
8.71.2.2 get_mem_stat(TMemotyUsage &memStat) . . . 283
8.72 uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH > Struct Template Reference . . . 283
8.72.1 Detailed Description . . . 284
8.72.2 Member Typedef Documentation . . . 284
8.72.2.1 covered_info . . . 284
8.72.2.2 state_frame . . . 285
8.72.3 Constructor & Destructor Documentation . . . 285
8.72.3.1 state_data_templ(const stack_data &stack_data) . . . 285
8.72.3.2 state_data_templ(const state_data_templ &prev_state_data) . . . 285
8.72.3.3 state_data_templ(const state_data_templ &prev_state_data, const int32_↩ t &begin_pos, const int32_t &end_pos, const covered_info &covered, tm_const↩ _target_entry *target) . . . 285
8.72.4 Member Function Documentation . . . 286
8.72.4.1 covered_to_string() const . . . 286
8.72.5 Member Data Documentation . . . 286
8.72.5.1 m_begin_lm_level . . . 286
8.72.5.2 m_covered . . . 286
8.72.5.3 m_partial_score . . . 286
8.72.5.4 m_s_begin_word_idx . . . 286
8.72.5.5 m_s_end_word_idx . . . 286

8.72.5.6 m_stack_data . . . 287
8.72.5.7 m_stack_level . . . 287
8.72.5.8 m_target . . . 287
8.72.5.9 m_total_score . . . 287
8.72.5.10 m_trans_frame . . . 287
8.72.5.11 rm_entry_data . . . 287
8.72.5.12 UNDEFINED_WORD_IDX . . . 287
8.72.5.13 ZERRO_WORD_IDX . . . 287
8.73 uva::smt::bpbd::server::lm::m_grams::m_gram_id::T_Gram_Id_Key Struct Reference . . . 288
8.73.1 Detailed Description . . . 288
8.73.2 Member Data Documentation . . . 288
8.73.2.1 m_id . . . 288
8.73.2.2 m_len_bytes . . . 288
8.74 uva::utils::containers::utils::T_IS_COMPARE_FUNC< ELEM_TYPE > Struct Template Reference 288
8.74.1 Detailed Description . . . 288
8.74.2 Member Typedef Documentation . . . 289
8.74.2.1 func_ptr . . . 289
8.74.2.2 func_type . . . 289
8.75 uva::smt::bpbd::server::lm::arpa::TAddGramFunct< WordIndexType > Struct Template Reference . 289
8.75.1 Detailed Description . . . 289
8.75.2 Member Typedef Documentation . . . 289
8.75.2.1 func . . . 289
8.76 uva::smt::bpbd::server::lm::__C2WArrayTrie::TCtxIdProbData Struct Reference . . . 289
8.76.1 Detailed Description . . . 290
8.76.2 Member Data Documentation . . . 290
8.76.2.1 ctx_id . . . 290
8.76.2.2 prob . . . 290
8.76.2.3 word_id . . . 290
8.77 uva::utils::file::text_piece_reader Class Reference . . . 290
8.77.1 Detailed Description . . . 291
8.77.2 Constructor & Destructor Documentation . . . 291
8.77.2.1 text_piece_reader() . . . 291
8.77.2.2 text_piece_reader(const void *begin_ptr, const size_t len) . . . 291
8.77.2.3 text_piece_reader(const text_piece_reader &other) . . . 291
8.77.3 Member Function Documentation . . . 292
8.77.3.1 copy_string(const text_piece_reader &other) . . . 292
8.77.3.2 find_first_subseq() . . . 292
8.77.3.3 get_begin_c_str() const . . . 292
8.77.3.4 get_begin_ptr() const . . . 292
8.77.3.5 get_first(text_piece_reader &out) . . . 292

8.77.3.6 get_first_line(text_piece_reader &out) . . . 293
8.77.3.7 get_first_space(text_piece_reader &out) . . . 293
8.77.3.8 get_first_tab(text_piece_reader &out) . . . 293
8.77.3.9 get_last(text_piece_reader &out) . . . 294
8.77.3.10 get_last_space(text_piece_reader &out) . . . 295
8.77.3.11 get_rest_c_str() const . . . 295
8.77.3.12 get_rest_str() const . . . 295
8.77.3.13 has_more() . . . 295
8.77.3.14 length() const . . . 296
8.77.3.15 operator"!=(const text_piece_reader &other) const . . . 296
8.77.3.16 operator"!=(const char *other) const . . . 296
8.77.3.17 operator"!=(const string &other) const . . . 296
8.77.3.18 operator==(const text_piece_reader &other) const . . . 296
8.77.3.19 operator==(const char *other) const . . . 296
8.77.3.20 operator==(const string &other) const . . . 297
8.77.3.21 operator[](size_t idx) . . . 297
8.77.3.22 set(const void *begin_ptr, const size_t len) . . . 297
8.77.3.23 str() const . . . 297
8.78 uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type > Class Template Reference . . . 297
8.78.1 Detailed Description . . . 298
8.78.2 Constructor & Destructor Documentation . . . 298
8.78.2.1 tm_basic_builder(const tm_parameters &params, model_type &model, reader_type &reader) . . . 298
8.78.2.2 ~tm_basic_builder() . . . 298
8.78.3 Member Function Documentation . . . 299
8.78.3.1 add_unk_translation() . . . 299
8.78.3.2 build() . . . 299
8.78.3.3 count_source_phrases() . . . 299
8.78.3.4 is_good_features(text_piece_reader rest, size_t &tmp_features_size, prob_weight *tmp_features) . . . 299
8.78.3.5 parse_tm_file() . . . 299
8.78.3.6 post_process_feature(const float feature, const float lambda) . . . 300
8.78.3.7 process_features(text_piece_reader weights, size_t &num_features, prob_weight *storage) . . . 300
8.78.3.8 process_source_entries() . . . 300
8.78.3.9 process_target_entry(tm_source_entry *source_entry, text_piece_reader &rest, size_t &count_ref, size_t &tmp_features_size, prob_weight *tmp_features) . . . 300
8.79 uva::smt::bpbd::server::tm::models::tm_basic_model Class Reference . . . 301
8.79.1 Detailed Description . . . 301
8.79.2 Member Typedef Documentation . . . 302

8.79.2.1 tm_source_entry_map . . . 302
8.79.3 Constructor & Destructor Documentation . . . 302
8.79.3.1 tm_basic_model() . . . 302
8.79.3.2 ~tm_basic_model() . . . 302
8.79.4 Member Function Documentation . . . 302
8.79.4.1 begin_entry(const phrase_uid entry_id, const size_t num_elems) . . . 302
8.79.4.2 finalize() . . . 302
8.79.4.3 finalize_entry(const phrase_uid entry_id) . . . 302
8.79.4.4 get_source_entry(const phrase_uid entry_id) const . . . 303
8.79.4.5 is_num_entries_needed() const . . . 304
8.79.4.6 is_unk_entry(tm_const_source_entry *entry) const . . . 304
8.79.4.7 log_model_type_info() const . . . 304
8.79.4.8 set_num_entries(const size_t num_entries) . . . 304
8.79.4.9 set_unk_entry(word_uid unk_word_id, const size_t num_unk_features, feature_array unk_features, const prob_weight lm_weight) . . . 305
8.80 uva::smt::bpbd::server::tm::tm_configurator Class Reference . . . 306
8.80.1 Detailed Description . . . 306
8.80.2 Member Function Documentation . . . 306
8.80.2.1 allocate_query_proxy() . . . 306
8.80.2.2 connect(const tm_parameters &params) . . . 306
8.80.2.3 disconnect() . . . 307
8.80.2.4 dispose_query_proxy(tm_query_proxy &query) . . . 307
8.81 uva::smt::bpbd::server::tm::tm_parameters Struct Reference . . . 307
8.81.1 Detailed Description . . . 307
8.81.2 Member Function Documentation . . . 308
8.81.2.1 finalize() . . . 308
8.81.3 Member Data Documentation . . . 308
8.81.3.1 m_conn_string . . . 308
8.81.3.2 m_lambdas . . . 308
8.81.3.3 m_min_tran_prob . . . 308
8.81.3.4 m_num_lambdas . . . 308
8.81.3.5 m_num_unk_features . . . 308
8.81.3.6 m_trans_limit . . . 308
8.81.3.7 m_unk_features . . . 308
8.82 uva::smt::bpbd::server::tm::proxy::tm_proxy Class Reference . . . 308
8.82.1 Detailed Description . . . 309
8.82.2 Constructor & Destructor Documentation . . . 309
8.82.2.1 ~tm_proxy() . . . 309
8.82.3 Member Function Documentation . . . 309
8.82.3.1 allocate_query_proxy()=0 . . . 309

8.82.3.2 connect(const tm_parameters &params)=0 . . . 309
8.82.3.3 disconnect()=0 . . . 310
8.82.3.4 dispose_query_proxy(tm_query_proxy &query)=0 . . . 310
8.83 uva::smt::bpbd::server::tm::proxy::tm_proxy_local Class Reference . . . 310
8.83.1 Detailed Description . . . 310
8.83.2 Constructor & Destructor Documentation . . . 311
8.83.2.1 tm_proxy_local() . . . 311
8.83.2.2 ∼tm_proxy_local() . . . 311
8.83.3 Member Function Documentation . . . 311
8.83.3.1 allocate_query_proxy() . . . 311
8.83.3.2 connect(const tm_parameters &params) . . . 311
8.83.3.3 disconnect() . . . 311
8.83.3.4 dispose_query_proxy(tm_query_proxy &query) . . . 311
8.83.3.5 load_model_data(char const ∗model_name, const tm_parameters &params) . . . 312
8.84 uva::smt::bpbd::server::tm::models::tm_query< model_type > Class Template Reference . . . 312
8.84.1 Detailed Description . . . 312
8.84.2 Member Typedef Documentation . . . 312
8.84.2.1 query_map . . . 312
8.84.3 Constructor & Destructor Documentation . . . 313
8.84.3.1 tm_query(const model_type &model) . . . 313
8.84.3.2 ∼tm_query() . . . 313
8.84.4 Member Function Documentation . . . 313
8.84.4.1 execute(const phrase_uid uid, tm_const_source_entry_ptr &entry_ptr) . . . 313
8.84.4.2 get_source_entry(const phrase_uid uid) . . . 313
8.84.4.3 get_st_uids(vector< phrase_uid > &st_uids) const . . . 313
8.85 uva::smt::bpbd::server::tm::proxy::tm_query_proxy Class Reference . . . 314
8.85.1 Detailed Description . . . 314
8.85.2 Constructor & Destructor Documentation . . . 314
8.85.2.1 ∼tm_query_proxy() . . . 314
8.85.3 Member Function Documentation . . . 314
8.85.3.1 execute(const phrase_uid uid, tm_const_source_entry_ptr &entry_ptr)=0 . . . 314
8.85.3.2 get_source_entry(const phrase_uid uid)=0 . . . 315
8.85.3.3 get_st_uids(vector< phrase_uid > &st_uids) const =0 . . . 315
8.86 uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local< model_type > Class Template Reference 315
8.86.1 Detailed Description . . . 316
8.86.2 Constructor & Destructor Documentation . . . 316
8.86.2.1 tm_query_proxy_local(const model_type &model) . . . 316
8.86.2.2 ∼tm_query_proxy_local() . . . 316
8.86.3 Member Function Documentation . . . 316
8.86.3.1 execute(const phrase_uid uid, tm_const_source_entry_ptr &entry_ptr) . . . 316

8.86.3.2 get_source_entry(const phrase_uid uid) . . . 316
8.86.3.3 get_st_uids(vector< phrase_uid > &st_uids) const . . . 317
8.87 uva::smt::bpbd::server::tm::models::tm_source_entry Class Reference . . . 317
8.87.1 Detailed Description . . . 317
8.87.2 Constructor & Destructor Documentation . . . 317
8.87.2.1 tm_source_entry() . . . 317
8.87.2.2 ~tm_source_entry() . . . 318
8.87.3 Member Function Documentation . . . 318
8.87.3.1 add_target(const string &target, const phrase_uid target_uid, const size_t num_features, const prob_weight *features, const phrase_length num_words, const word_uid *word_ids, const prob_weight lm_weight) . . . 318
8.87.3.2 begin(const size_t capacity) . . . 318
8.87.3.3 finalize() . . . 318
8.87.3.4 get_min_cost() const . . . 318
8.87.3.5 get_source_uid() . . . 318
8.87.3.6 get_st_uids(vector< phrase_uid > &st_uids) const . . . 319
8.87.3.7 get_targets() const . . . 319
8.87.3.8 has_target(const phrase_uid target_uid) const . . . 319
8.87.3.9 has_translations() const . . . 319
8.87.3.10 num_targets() const . . . 320
8.87.3.11 operator==(const phrase_uid &phrase_uid) const . . . 320
8.87.3.12 operator==(const tm_source_entry &other) const . . . 320
8.87.3.13 set_source_uid(phrase_uid s_uid) . . . 320
8.88 uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features > Class Template Reference . . . 320
8.88.1 Detailed Description . . . 321
8.88.2 Constructor & Destructor Documentation . . . 321
8.88.2.1 tm_target_entry_temp() . . . 321
8.88.2.2 ~tm_target_entry_temp() . . . 321
8.88.3 Member Function Documentation . . . 322
8.88.3.1 get_num_words() const . . . 322
8.88.3.2 get_st_uid() const . . . 322
8.88.3.3 get_t_c_s() const . . . 322
8.88.3.4 get_target_phrase() const . . . 322
8.88.3.5 get_total_weight() const . . . 322
8.88.3.6 get_word_ids() const . . . 323
8.88.3.7 is_unk_trans() const . . . 323
8.88.3.8 set_data(const phrase_uid source_uid, const string &target_phrase, const phrase_uid target_uid, const size_t num_features, const float *features, const phrase_length num_words, const word_uid *word_ids) . . . 323
8.88.3.9 set_features(const size_t num_features, const prob_weight *features) . . . 323

8.88.4 Member Data Documentation . . . 324
8.88.4.1 NUM_FEATURES . . . 324
8.88.4.2 UNKNOWN_TARGET_ENTRY_UID . . . 324
8.89 uva::smt::bpbd::client::trans_job Struct Reference . . . 324
8.89.1 Detailed Description . . . 324
8.89.2 Constructor & Destructor Documentation . . . 325
8.89.2.1 trans_job() . . . 325
8.89.2.2 ~trans_job() . . . 325
8.89.3 Member Data Documentation . . . 325
8.89.3.1 m_num_sentences . . . 325
8.89.3.2 m_request . . . 325
8.89.3.3 m_response . . . 325
8.89.3.4 m_status . . . 325
8.90 uva::smt::bpbd::server::trans_job Class Reference . . . 325
8.90.1 Detailed Description . . . 326
8.90.2 Member Typedef Documentation . . . 326
8.90.2.1 done_job_notifier . . . 326
8.90.2.2 tasks_const_iter_type . . . 326
8.90.2.3 tasks_iter_type . . . 326
8.90.2.4 tasks_list_type . . . 326
8.90.3 Constructor & Destructor Documentation . . . 326
8.90.3.1 trans_job(trans_job_request_ptr request_ptr) . . . 326
8.90.3.2 ~trans_job() . . . 327
8.90.4 Member Function Documentation . . . 327
8.90.4.1 cancel() . . . 327
8.90.4.2 combine_job_result() . . . 327
8.90.4.3 get_code() const . . . 327
8.90.4.4 get_job_id() const . . . 327
8.90.4.5 get_session_id() const . . . 327
8.90.4.6 get_tasks() . . . 328
8.90.4.7 get_text() const . . . 328
8.90.4.8 is_job_finished() . . . 328
8.90.4.9 notify_task_done(const trans_task_ptr &task) . . . 328
8.90.4.10 set_done_job_notifier(done_job_notifier notify_job_done_func) . . . 328
8.91 uva::smt::bpbd::common::messaging::trans_job_code Class Reference . . . 329
8.91.1 Detailed Description . . . 329
8.91.2 Member Enumeration Documentation . . . 329
8.91.2.1 values . . . 329
8.91.3 Constructor & Destructor Documentation . . . 329
8.91.3.1 trans_job_code(const values code) . . . 329

8.91.3.2 trans_job_code(const int32_t code_val) . . . 330
8.91.3.3 trans_job_code() . . . 330
8.91.4 Member Function Documentation . . . 330
8.91.4.1 operator int() const . . . 330
8.91.4.2 operator string() const . . . 330
8.91.4.3 operator<(const values &code) const . . . 330
8.91.4.4 operator=(const values &code) . . . 330
8.91.4.5 operator==(const values &code) const . . . 331
8.91.4.6 str() const . . . 331
8.91.4.7 val() . . . 331
8.92 uva::smt::bpbd::server::trans_job_pool Class Reference . . . 331
8.92.1 Detailed Description . . . 332
8.92.2 Member Typedef Documentation . . . 332
8.92.2.1 finished_job_notifier . . . 332
8.92.2.2 jobs_list_iter_type . . . 332
8.92.2.3 jobs_list_type . . . 332
8.92.2.4 jobs_map_iter_type . . . 332
8.92.2.5 jobs_map_type . . . 332
8.92.2.6 sessions_map_iter_type . . . 333
8.92.2.7 sessions_map_type . . . 333
8.92.3 Constructor & Destructor Documentation . . . 333
8.92.3.1 trans_job_pool(const size_t num_threads) . . . 333
8.92.3.2 ~trans_job_pool() . . . 333
8.92.4 Member Function Documentation . . . 333
8.92.4.1 add_job(trans_job_ptr trans_job) . . . 333
8.92.4.2 cancel_all_jobs() . . . 333
8.92.4.3 cancel_jobs(const session_id_type session_id) . . . 333
8.92.4.4 delete_job(trans_job_ptr trans_job) . . . 334
8.92.4.5 is_stop_running() . . . 334
8.92.4.6 notify_job_done(trans_job_ptr trans_job) . . . 334
8.92.4.7 plan_new_job(trans_job_ptr trans_job) . . . 334
8.92.4.8 process_finished_jobs() . . . 334
8.92.4.9 report_run_time_info() . . . 334
8.92.4.10 set_job_result_setter(finished_job_notifier notify_job_finished_func) . . . 335
8.92.4.11 set_num_threads(const size_t num_threads) . . . 336
8.92.4.12 stop() . . . 336
8.92.4.13 wake_up_jobs_thread() . . . 336
8.93 uva::smt::bpbd::common::messaging::trans_job_request Class Reference . . . 336
8.93.1 Detailed Description . . . 337
8.93.2 Constructor & Destructor Documentation . . . 337

8.93.2.1 trans_job_request(const string &message) . . . 337
8.93.2.2 trans_job_request(const job_id_type job_id, const string &source_lang, const string &text, const string &target_lang) . . . 337
8.93.3 Member Function Documentation . . . 337
8.93.3.1 de_serialize(const string &message) . . . 337
8.93.3.2 get_job_id() const . . . 337
8.93.3.3 get_session_id() const . . . 338
8.93.3.4 get_source_lang() const . . . 338
8.93.3.5 get_target_lang() const . . . 338
8.93.3.6 get_text() const . . . 338
8.93.3.7 serialize() const . . . 338
8.93.3.8 set_session_id(const session_id_type session_id) . . . 338
8.93.4 Member Data Documentation . . . 339
8.93.4.1 HEADER_DELIMITER . . . 339
8.93.4.2 NEW_LINE_HEADER_ENDING . . . 339
8.93.4.3 TEXT_SENTENCE_DELIMITER . . . 339
8.94 uva::smt::bpbd::common::messaging::trans_job_response Class Reference . . . 339
8.94.1 Detailed Description . . . 339
8.94.2 Constructor & Destructor Documentation . . . 340
8.94.2.1 trans_job_response() . . . 340
8.94.2.2 trans_job_response(const string &message) . . . 340
8.94.2.3 trans_job_response(const job_id_type job_id, const trans_job_code code, const string &text) . . . 340
8.94.3 Member Function Documentation . . . 340
8.94.3.1 de_serialize(const string &message) . . . 340
8.94.3.2 get_code() const . . . 340
8.94.3.3 get_job_id() const . . . 341
8.94.3.4 get_text() const . . . 341
8.94.3.5 is_good() const . . . 341
8.94.3.6 is_job_id_defined() const . . . 341
8.94.3.7 serialize() . . . 341
8.94.4 Member Data Documentation . . . 341
8.94.4.1 HEADER_DELIMITER . . . 341
8.94.4.2 NEW_LINE_HEADER_ENDING . . . 342
8.95 uva::smt::bpbd::client::trans_job_status Class Reference . . . 342
8.95.1 Detailed Description . . . 342
8.95.2 Member Enumeration Documentation . . . 342
8.95.2.1 values . . . 342
8.95.3 Constructor & Destructor Documentation . . . 343
8.95.3.1 trans_job_status(const values status) . . . 343

8.95.3.2 trans_job_status(const int32_t status_val) . . . 343
8.95.3.3 trans_job_status() . . . 343
8.95.4 Member Function Documentation . . . 343
8.95.4.1 operator int() const . . . 343
8.95.4.2 operator string() const . . . 343
8.95.4.3 operator<(const values &status) const . . . 343
8.95.4.4 operator=(const values &status) . . . 344
8.95.4.5 operator==(const values &status) const . . . 344
8.95.4.6 str() const . . . 344
8.96 uva::smt::bpbd::server::trans_manager Class Reference . . . 344
8.96.1 Detailed Description . . . 345
8.96.2 Member Typedef Documentation . . . 345
8.96.2.1 handlers_map_iter_type . . . 345
8.96.2.2 handlers_map_type . . . 345
8.96.2.3 response_sender . . . 345
8.96.2.4 sessions_map_type . . . 345
8.96.3 Constructor & Destructor Documentation . . . 345
8.96.3.1 trans_manager(const size_t num_threads) . . . 345
8.96.3.2 ~trans_manager() . . . 346
8.96.4 Member Function Documentation . . . 346
8.96.4.1 close_session(websocketpp::connection_hdl hdl) . . . 346
8.96.4.2 notify_job_finished(trans_job_ptr trans_job) . . . 346
8.96.4.3 open_session(websocketpp::connection_hdl hdl) . . . 346
8.96.4.4 report_run_time_info() . . . 346
8.96.4.5 set_num_threads(const size_t num_threads) . . . 347
8.96.4.6 set_response_sender(response_sender sender) . . . 347
8.96.4.7 stop() . . . 347
8.96.4.8 translate(websocketpp::connection_hdl hdl, trans_job_request_ptr request_ptr) . . . 347
8.97 uva::smt::bpbd::client::trans_manager Class Reference . . . 347
8.97.1 Detailed Description . . . 348
8.97.2 Member Typedef Documentation . . . 348
8.97.2.1 jobs_list_iter_type . . . 348
8.97.2.2 jobs_list_type . . . 348
8.97.2.3 jobs_map_iter_type . . . 348
8.97.2.4 jobs_map_type . . . 348
8.97.3 Constructor & Destructor Documentation . . . 349
8.97.3.1 trans_manager(const client_config &params) . . . 349
8.97.3.2 ~trans_manager() . . . 349
8.97.4 Member Function Documentation . . . 349
8.97.4.1 check_jobs_done_and_notify() . . . 349

8.97.4.2 get_num_of_sentences() . . . 349
8.97.4.3 notify_conn_closed() . . . 349
8.97.4.4 notify_jobs_done() . . . 349
8.97.4.5 notify_jobs_sent() . . . 349
8.97.4.6 send_translation_jobs() . . . 350
8.97.4.7 set_job_response(trans_job_response *trans_job_resp) . . . 350
8.97.4.8 start() . . . 350
8.97.4.9 stop() . . . 350
8.97.4.10 wait() . . . 350
8.97.4.11 write_received_job_result(const uint32_t fis, const uint32_t lis, const trans_job_ptr job, ofstream &target_file) . . . 350
8.97.4.12 write_result_to_file() . . . 350
8.97.5 Member Data Documentation . . . 350
8.97.5.1 MIN_SENTENCES_PER_REQUEST . . . 351
8.98 uva::smt::bpbd::server::trans_task Class Reference . . . 351
8.98.1 Detailed Description . . . 351
8.98.2 Member Typedef Documentation . . . 351
8.98.2.1 cancel_task_notifier . . . 351
8.98.2.2 done_task_notifier . . . 351
8.98.3 Constructor & Destructor Documentation . . . 352
8.98.3.1 trans_task(const session_id_type session_id, const job_id_type job_id, const task_id_type task_id, const string &source_sentence, done_task_notifier notify_task_done_func) . . . 352
8.98.3.2 ~trans_task() . . . 353
8.98.4 Member Function Documentation . . . 353
8.98.4.1 cancel() . . . 353
8.98.4.2 get_code() const . . . 353
8.98.4.3 get_source_text() const . . . 353
8.98.4.4 get_target_text() . . . 353
8.98.4.5 get_task_id() const . . . 354
8.98.4.6 process_task_result() . . . 354
8.98.4.7 set_cancel_task_notifier(cancel_task_notifier notify_task_cancel_func) . . . 354
8.98.4.8 translate() . . . 354
8.99 uva::smt::bpbd::server::trans_task_pool Class Reference . . . 354
8.99.1 Detailed Description . . . 355
8.99.2 Member Typedef Documentation . . . 355
8.99.2.1 tasks_queue_iter_type . . . 355
8.99.2.2 tasks_queue_type . . . 355
8.99.2.3 threads_list_type . . . 355
8.99.2.4 workers_list_type . . . 355
8.99.3 Constructor & Destructor Documentation . . . 355

8.99.3.1 trans_task_pool(const size_t num_threads) . . . 355
8.99.3.2 ~trans_task_pool() . . . 356
8.99.4 Member Function Documentation . . . 356
8.99.4.1 notify_task_cancel(trans_task_ptr trans_task) . . . 356
8.99.4.2 plan_new_task(trans_task_ptr trans_task) . . . 356
8.99.4.3 report_run_time_info() . . . 356
8.99.4.4 set_num_threads(const size_t num_threads) . . . 356
8.99.5 Friends And Related Function Documentation . . . 357
8.99.5.1 trans_task_pool_worker . . . 357
8.99.6 Member Data Documentation . . . 357
8.99.6.1 m_condition . . . 357
8.99.6.2 m_queue_mutex . . . 357
8.99.6.3 m_stop . . . 357
8.99.6.4 m_tasks . . . 357
8.100 uva::smt::bpbd::server::trans_task_pool_worker Class Reference . . . 357
8.100.1 Detailed Description . . . 357
8.100.2 Constructor & Destructor Documentation . . . 358
8.100.2.1 trans_task_pool_worker(trans_task_pool &pool) . . . 358
8.100.2.2 ~trans_task_pool_worker() . . . 358
8.100.3 Member Function Documentation . . . 358
8.100.3.1 is_busy() . . . 358
8.100.3.2 operator()() . . . 358
8.100.3.3 stop() . . . 358
8.101 uva::smt::bpbd::client::translation_client Class Reference . . . 358
8.101.1 Detailed Description . . . 359
8.101.2 Member Typedef Documentation . . . 359
8.101.2.1 client . . . 359
8.101.2.2 conn_close_notifier . . . 359
8.101.2.3 response_setter . . . 359
8.101.3 Constructor & Destructor Documentation . . . 359
8.101.3.1 translation_client(const string &host, const uint16_t port, response_setter set_response, conn_close_notifier notify_conn_close) . . . 359
8.101.3.2 ~translation_client() . . . 360
8.101.4 Member Function Documentation . . . 360
8.101.4.1 connect() . . . 360
8.101.4.2 disconnect() . . . 360
8.101.4.3 get_uri() . . . 360
8.101.4.4 on_close(websocketpp::connection_hdl hdl) . . . 360
8.101.4.5 on_fail(websocketpp::connection_hdl hdl) . . . 360
8.101.4.6 on_message(websocketpp::connection_hdl hdl, client::message_ptr msg) . . . 361

8.101.4.7 on_open(websocketpp::connection_hdl hdl) . . . 361
8.101.4.8 send(const trans_job_request_ptr request) . . . 361
8.101.4.9 wait_connect() . . . 361
8.102 uva::smt::bpbd::server::translation_server Class Reference . . . 361
8.102.1 Detailed Description . . . 362
8.102.2 Member Typedef Documentation . . . 362
8.102.2.1 server . . . 362
8.102.3 Constructor & Destructor Documentation . . . 362
8.102.3.1 translation_server(const uint16_t port, const size_t num_threads) . . . 362
8.102.4 Member Function Documentation . . . 362
8.102.4.1 on_close(connection_hdl hdl) . . . 362
8.102.4.2 on_fail(connection_hdl hdl) . . . 363
8.102.4.3 on_message(websocketpp::connection_hdl hdl, server::message_ptr msg) . . . 363
8.102.4.4 on_open(connection_hdl hdl) . . . 363
8.102.4.5 report_run_time_info() . . . 363
8.102.4.6 run() . . . 363
8.102.4.7 send_response(connection_hdl hdl, trans_job_response &response) . . . 363
8.102.4.8 set_num_threads(const size_t num_threads) . . . 364
8.102.4.9 stop() . . . 364
8.103 uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::TSubArrReference Struct Reference 364
8.103.1 Detailed Description . . . 364
8.103.2 Member Data Documentation . . . 364
8.103.2.1 begin_idx . . . 365
8.103.2.2 end_idx . . . 365
8.104 uva::smt::bpbd::server::lm::__C2WArrayTrie::TWordIdPBData Struct Reference . . . 365
8.104.1 Detailed Description . . . 365
8.104.2 Member Data Documentation . . . 365
8.104.2.1 id . . . 365
8.104.2.2 payload . . . 365
8.105 uva::smt::bpbd::server::lm::dictionary::__counting_word_index::TWordInfo Struct Reference . . . 365
8.105.1 Detailed Description . . . 366
8.105.2 Member Data Documentation . . . 366
8.105.2.1 prob . . . 366
8.105.2.2 word . . . 366
8.106 uva::utils::containers::upp_diag_matrix< element_type > Class Template Reference . . . 366
8.106.1 Detailed Description . . . 367
8.106.2 Member Typedef Documentation . . . 367
8.106.2.1 element_type_ptr . . . 367
8.106.3 Constructor & Destructor Documentation . . . 367
8.106.3.1 upp_diag_matrix(const size_t dim) . . . 367

8.106.3.2 ∼upp_diag_matrix() . . . 367
8.106.4 Member Function Documentation . . . 367
8.106.4.1 get_dim() const . . . 367
8.106.4.2 operator[](size_t idx) const . . . 367
8.106.5 Member Data Documentation . . . 368
8.106.5.1 m_max_idx . . . 368
8.106.5.2 m_min_idx . . . 368
8.107uva::utils::exceptions::uva_exception Class Reference . . . 368
8.107.1 Detailed Description . . . 368
8.107.2 Constructor & Destructor Documentation . . . 369
8.107.2.1 uva_exception(const char ∗message) . . . 369
8.107.2.2 uva_exception(const string &message) . . . 369
8.107.2.3 uva_exception(uva_exception const &other) . . . 369
8.107.2.4 ∼uva_exception() . . . 369
8.107.3 Member Function Documentation . . . 369
8.107.3.1 get_message() const . . . 369
8.107.3.2 what() const . . . 369
8.108uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType > Class Template Reference . . . 370
8.108.1 Detailed Description . . . 371
8.108.2 Member Typedef Documentation . . . 371
8.108.2.1 BASE . . . 371
8.108.2.2 T_M_GramWordEntry . . . 371
8.108.2.3 T_N_GramWordEntry . . . 371
8.108.3 Constructor & Destructor Documentation . . . 371
8.108.3.1 w2c_array_trie(WordIndexType &word_index) . . . 371
8.108.3.2 ∼w2c_array_trie() . . . 371
8.108.4 Member Function Documentation . . . 372
8.108.4.1 add_m_gram(const model_m_gram &gram) . . . 372
8.108.4.2 get_ctx_id(const phrase_length level_idx, const TShortId word_id, TLongId &ctx_id) const . . . 372
8.108.4.3 get_m_gram_payload(m_gram_query &query, MGramStatusEnum &status) const . . . 372
8.108.4.4 get_n_gram_payload(m_gram_query &query, MGramStatusEnum &status) const 372
8.108.4.5 get_unigram_payload(m_gram_query &query) const . . . 372
8.108.4.6 get_unk_word_prob() const . . . 373
8.108.4.7 is_post_grams() const . . . 373
8.108.4.8 log_model_type_info() const . . . 373
8.108.4.9 post_grams() . . . 373
8.108.4.10post_m_grams() . . . 373
8.108.4.11post_M_N_Grams(WORD_ENTRY_TYPE ∗wordsArray) . . . 374

8.108.4.12 post_n_grams() . . . 374
8.108.4.13 pre_allocate(const size_t counts[LM_M_GRAM_LEVEL_MAX]) . . . 374
8.108.4.14 set_def_unk_word_prob(const prob_weight prob) . . . 374
8.109 uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer > Class Template Reference . . . 374
8.109.1 Detailed Description . . . 375
8.109.2 Member Typedef Documentation . . . 375
8.109.2.1 BASE . . . 376
8.109.3 Constructor & Destructor Documentation . . . 376
8.109.3.1 w2c_hybrid_trie(WordIndexType &word_index) . . . 376
8.109.3.2 ~w2c_hybrid_trie() . . . 376
8.109.4 Member Function Documentation . . . 376
8.109.4.1 add_m_gram(const model_m_gram &gram) . . . 376
8.109.4.2 get_ctx_id(const phrase_length level_idx, const TShortId word_id, TLongId &ctx_id) const . . . 376
8.109.4.3 get_m_gram_payload(m_gram_query &query, MGramStatusEnum &status) const . . . 377
8.109.4.4 get_n_gram_payload(m_gram_query &query, MGramStatusEnum &status) const 377
8.109.4.5 get_unigram_payload(m_gram_query &query) const . . . 377
8.109.4.6 get_unk_word_prob() const . . . 377
8.109.4.7 log_model_type_info() const . . . 378
8.109.4.8 pre_allocate(const size_t counts[LM_M_GRAM_LEVEL_MAX]) . . . 378
8.109.4.9 set_def_unk_word_prob(const prob_weight prob) . . . 378
8.110 uva::smt::bpbd::server::lm::W2CH_UM_Storage Class Reference . . . 378
8.110.1 Detailed Description . . . 379
8.110.2 Member Typedef Documentation . . . 379
8.110.2.1 const_iterator . . . 379
8.110.3 Constructor & Destructor Documentation . . . 379
8.110.3.1 W2CH_UM_Storage(TStorageMapAllocator &alloc) . . . 379
8.110.3.2 ~W2CH_UM_Storage() . . . 379
8.110.4 Member Function Documentation . . . 379
8.110.4.1 at(const TShortId ctx_idx) const . . . 379
8.110.4.2 end() . . . 379
8.110.4.3 find(const TShortId ctx_idx) . . . 379
8.110.4.4 operator[](const TShortId ctx_idx) . . . 379
8.111 uva::smt::bpbd::server::lm::W2CH_UM_StorageFactory< N > Class Template Reference . . . 379
8.111.1 Detailed Description . . . 380
8.111.2 Constructor & Destructor Documentation . . . 380
8.111.2.1 W2CH_UM_StorageFactory(const size_t _counts[N], const float factor=__W2C↩ HybridTrie::UM_CTX_TO_PB_MAP_STORE_MEMORY_FACTOR) . . . 380
8.111.2.2 ~W2CH_UM_StorageFactory() . . . 380

8.111.3 Member Function Documentation . . . 380
8.111.3.1 create(const phrase_length level) . . . 380
8.111.4 Member Data Documentation . . . 381
8.111.4.1 m_p_alloc . . . 381
8.112 uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index::word_index_bucket_entry< word_id_type > Struct Template Reference . . . 381
8.112.1 Detailed Description . . . 381
8.112.2 Constructor & Destructor Documentation . . . 381
8.112.2.1 word_index_bucket_entry() . . . 381
8.112.3 Member Data Documentation . . . 382
8.112.3.1 m_len . . . 382
8.112.3.2 m_word . . . 382
8.112.3.3 m_word_id . . . 382
8.113 uva::smt::bpbd::server::lm::word_index_trie_base< WordIndex > Class Template Reference . . . 382
8.113.1 Detailed Description . . . 382
8.113.2 Member Typedef Documentation . . . 383
8.113.2.1 WordIndexType . . . 383
8.113.3 Constructor & Destructor Documentation . . . 383
8.113.3.1 word_index_trie_base(WordIndexType &word_index) . . . 383
8.113.4 Member Function Documentation . . . 383
8.113.4.1 get_word_index() const . . . 383
8.113.4.2 is_post_grams() const . . . 383
8.113.4.3 post_grams() . . . 383
8.113.4.4 pre_allocate(const size_t counts[LM_M_GRAM_LEVEL_MAX]) . . . 384
8.113.4.5 set_def_unk_word_prob(const prob_weight prob) . . . 384
8.113.5 Member Data Documentation . . . 384
8.113.5.1 m_word_index . . . 384
8.114 uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::WordDataEntry< ARRAY_ELEM_TYPE > Class Template Reference . . . 384
8.114.1 Detailed Description . . . 385
8.114.2 Member Data Documentation . . . 385
8.114.2.1 cio . . . 385

9 File Documentation **387**

9.1 inc/client/client_config.hpp File Reference . . . 387
9.2 inc/client/trans_job.hpp File Reference . . . 387
9.3 inc/server/trans_job.hpp File Reference . . . 388
9.4 inc/client/trans_job_status.hpp File Reference . . . 388
9.5 inc/client/trans_manager.hpp File Reference . . . 389
9.6 inc/server/trans_manager.hpp File Reference . . . 389
9.7 inc/client/translation_client.hpp File Reference . . . 390

9.7.1 Macro Definition Documentation . . . 390
9.7.1.1 ASIO_STANDALONE . . . 390
9.8 inc/common/messaging/id_manager.hpp File Reference . . . 390
9.9 inc/common/messaging/trans_job_code.hpp File Reference . . . 391
9.10 inc/common/messaging/trans_job_id.hpp File Reference . . . 391
9.11 inc/common/messaging/trans_job_request.hpp File Reference . . . 392
9.12 inc/common/messaging/trans_job_response.hpp File Reference . . . 392
9.13 inc/common/messaging/trans_session_id.hpp File Reference . . . 393
9.14 inc/common/utils/containers/array_utils.hpp File Reference . . . 393
9.14.1 Macro Definition Documentation . . . 394
9.14.1.1 BSEARCH_ONE_FIELD . . . 394
9.14.1.2 BSEARCH_TWO_FIELDS . . . 394
9.14.1.3 DECLARE_STATIC_BSEARCH_ID_FIELD_COMPARE_FUNC . . . 395
9.15 inc/common/utils/containers/circular_queue.hpp File Reference . . . 395
9.16 inc/common/utils/containers/dynamic_memory_arrays.hpp File Reference . . . 396
9.16.1 Macro Definition Documentation . . . 397
9.16.1.1 EXTRACT_C . . . 397
9.16.1.2 EXTRACT_P . . . 397
9.16.1.3 EXTRACT_PC . . . 397
9.16.1.4 EXTRACT_PCS . . . 397
9.16.1.5 EXTRACT_PS . . . 397
9.16.1.6 EXTRACT_S . . . 398
9.17 inc/common/utils/containers/fixed_size_hashmap.hpp File Reference . . . 398
9.18 inc/common/utils/containers/greedy_memory_allocator.hpp File Reference . . . 398
9.19 inc/common/utils/containers/greedy_memory_storage.hpp File Reference . . . 399
9.20 inc/common/utils/containers/upp_diag_matrix.hpp File Reference . . . 399
9.21 inc/common/utils/exceptions.hpp File Reference . . . 400
9.21.1 Macro Definition Documentation . . . 400
9.21.1.1 ASSERT_CONDITION_THROW . . . 400
9.21.1.2 ASSERT_SANITY_THROW . . . 400
9.21.1.3 THROW_EXCEPTION . . . 401
9.21.1.4 THROW_MUST_NOT_CALL . . . 401
9.21.1.5 THROW_MUST_OVERRIDE . . . 401
9.21.1.6 THROW_NOT_IMPLEMENTED . . . 401
9.22 inc/common/utils/file/afile_reader.hpp File Reference . . . 401
9.23 inc/common/utils/file/cstyle_file_reader.hpp File Reference . . . 401
9.24 inc/common/utils/file/file_stream_reader.hpp File Reference . . . 402
9.25 inc/common/utils/file/memory_mapped_file_reader.hpp File Reference . . . 402
9.26 inc/common/utils/file/text_piece_reader.hpp File Reference . . . 403
9.27 inc/common/utils/hashing_utils.hpp File Reference . . . 404

9.27.1 Macro Definition Documentation . . . 404
9.27.1.1 A . . . 404
9.27.1.2 B . . . 404
9.27.1.3 C . . . 404
9.27.1.4 cwfold . . . 405
9.27.1.5 cwmixa . . . 405
9.27.1.6 cwmixb . . . 405
9.27.1.7 get16bits . . . 405
9.28 inc/common/utils/logging/logger.hpp File Reference . . . 405
9.28.1 Macro Definition Documentation . . . 406
9.28.1.1 __FILENAME__ . . . 406
9.28.1.2 DEBUG1_PARAM_VALUE . . . 407
9.28.1.3 DEBUG2_PARAM_VALUE . . . 407
9.28.1.4 DEBUG3_PARAM_VALUE . . . 407
9.28.1.5 DEBUG4_PARAM_VALUE . . . 407
9.28.1.6 DEBUG_PARAM_VALUE . . . 407
9.28.1.7 END_LOG . . . 407
9.28.1.8 ERROR_PARAM_VALUE . . . 407
9.28.1.9 INFO1_PARAM_VALUE . . . 407
9.28.1.10 INFO2_PARAM_VALUE . . . 407
9.28.1.11 INFO3_PARAM_VALUE . . . 407
9.28.1.12 INFO_PARAM_VALUE . . . 407
9.28.1.13 LINE_STRING . . . 408
9.28.1.14 LOG_DEBUG . . . 408
9.28.1.15 LOG_DEBUG1 . . . 408
9.28.1.16 LOG_DEBUG2 . . . 408
9.28.1.17 LOG_DEBUG3 . . . 408
9.28.1.18 LOG_DEBUG4 . . . 408
9.28.1.19 LOG_ERROR . . . 408
9.28.1.20 LOG_INFO . . . 408
9.28.1.21 LOG_INFO1 . . . 408
9.28.1.22 LOG_INFO2 . . . 408
9.28.1.23 LOG_INFO3 . . . 408
9.28.1.24 LOG_RESULT . . . 408
9.28.1.25 LOG_USAGE . . . 409
9.28.1.26 LOG_WARNING . . . 409
9.28.1.27 LOGGER . . . 409
9.28.1.28 LOGGER_DEBUG . . . 409
9.28.1.29 PROGRESS_UPDATE_PERIOD . . . 409
9.28.1.30 RESULT_PARAM_VALUE . . . 409

9.28.1.31 SSTR . . . 409
9.28.1.32 STRINGIZE . . . 409
9.28.1.33 STRINGIZE2 . . . 409
9.28.1.34 USAGE_PARAM_VALUE . . . 410
9.28.1.35 WARNING_PARAM_VALUE . . . 410
9.28.1.36 WHITE_SPACE_SEPARATOR . . . 410
9.29 inc/common/utils/math_utils.hpp File Reference . . . 410
9.29.1 Macro Definition Documentation . . . 411
9.29.1.1 BYTE_IDX . . . 411
9.29.1.2 BYTES_TO_BITS . . . 411
9.29.1.3 HANDLE_ENDIAN . . . 411
9.29.1.4 NUM_BITS_REMAINDER . . . 411
9.29.1.5 NUM_BYTES_4_BITS . . . 411
9.29.1.6 NUM_FULL_BYTES . . . 411
9.29.1.7 REMAINING_BIT_IDX . . . 411
9.29.1.8 VALUE_LEN_BYTES . . . 411
9.30 inc/common/utils/monitor/statistics_monitor.hpp File Reference . . . 411
9.31 inc/common/utils/string_utils.hpp File Reference . . . 412
9.31.1 Macro Definition Documentation . . . 413
9.31.1.1 valid_digit . . . 413
9.32 inc/common/utils/threads.hpp File Reference . . . 413
9.33 inc/main.hpp File Reference . . . 413
9.33.1 Macro Definition Documentation . . . 414
9.33.1.1 GET_ASSERT . . . 414
9.33.1.2 MAX_STACK_TRACE_LEN . . . 414
9.33.1.3 SAFE_DESTROY . . . 414
9.34 inc/server/cmd_line_handler.hpp File Reference . . . 414
9.35 inc/server/common/models/phrase_uid.hpp File Reference . . . 415
9.36 inc/server/decoder/de_configs.hpp File Reference . . . 415
9.37 inc/server/decoder/de_configurator.hpp File Reference . . . 416
9.38 inc/server/decoder/de_parameters.hpp File Reference . . . 416
9.39 inc/server/decoder/sentence/sentence_data_map.hpp File Reference . . . 416
9.40 inc/server/decoder/sentence/sentence_decoder.hpp File Reference . . . 417
9.41 inc/server/decoder/stack/multi_stack.hpp File Reference . . . 418
9.42 inc/server/decoder/stack/stack_data.hpp File Reference . . . 418
9.43 inc/server/decoder/stack/stack_level.hpp File Reference . . . 419
9.44 inc/server/decoder/stack/stack_state.hpp File Reference . . . 419
9.45 inc/server/decoder/stack/state_data.hpp File Reference . . . 420
9.46 inc/server/lm/builders/lm_basic_builder.hpp File Reference . . . 420
9.47 inc/server/lm/builders/lm_gram_builder.hpp File Reference . . . 421

9.48 inc/server/lm/builders/lm_gram_builder_factory.hpp File Reference . . . 421
9.49 inc/server/lm/dictionaries/aword_index.hpp File Reference . . . 422
9.50 inc/server/lm/dictionaries/basic_word_index.hpp File Reference . . . 422
9.51 inc/server/lm/dictionaries/counting_word_index.hpp File Reference . . . 423
9.52 inc/server/lm/dictionaries/hashing_word_index.hpp File Reference . . . 423
9.53 inc/server/lm/dictionaries/optimizing_word_index.hpp File Reference . . . 424
9.53.1 Macro Definition Documentation . . . 425
9.53.1.1 IS_EQUAL . . . 425
9.53.2 Function Documentation . . . 425
9.53.2.1 word_index_bucket_entry() . . . 425
9.53.3 Variable Documentation . . . 425
9.53.3.1 m_len . . . 425
9.53.3.2 m_word . . . 425
9.53.3.3 m_word_id . . . 425
9.54 inc/server/lm/lm_configs.hpp File Reference . . . 426
9.55 inc/server/lm/lm_configurator.hpp File Reference . . . 426
9.56 inc/server/lm/lm_consts.hpp File Reference . . . 427
9.57 inc/server/lm/lm_executor.hpp File Reference . . . 428
9.58 inc/server/lm/lm_parameters.hpp File Reference . . . 428
9.59 inc/server/lm/mgrams/m_gram_id.hpp File Reference . . . 429
9.59.1 Macro Definition Documentation . . . 429
9.59.1.1 DECLARE_STACK_GRAM_ID . . . 429
9.59.1.2 MAX_N_GRAM_ID_LEN_BYTES . . . 429
9.59.1.3 N_GRAM_ID_TYPE_LEN_BYTES . . . 430
9.60 inc/server/lm/mgrams/m_gram_id_tables.hpp File Reference . . . 430
9.60.1 Macro Definition Documentation . . . 430
9.60.1.1 BYTE_M_GRAM_ID_TABLES_HPP . . . 430
9.61 inc/server/lm/mgrams/m_gram_payload.hpp File Reference . . . 430
9.62 inc/server/lm/mgrams/model_m_gram.hpp File Reference . . . 431
9.63 inc/server/lm/mgrams/query_m_gram.hpp File Reference . . . 431
9.64 inc/server/lm/models/bitmap_hash_cache.hpp File Reference . . . 432
9.65 inc/server/lm/models/c2d_hybrid_trie.hpp File Reference . . . 432
9.66 inc/server/lm/models/c2d_map_trie.hpp File Reference . . . 433
9.67 inc/server/lm/models/c2w_array_trie.hpp File Reference . . . 433
9.68 inc/server/lm/models/g2d_map_trie.hpp File Reference . . . 434
9.69 inc/server/lm/models/generic_trie_base.hpp File Reference . . . 435
9.69.1 Macro Definition Documentation . . . 436
9.69.1.1 INSTANTIATE_TRIE_FUNCS_LEVEL . . . 436
9.69.1.2 INSTANTIATE_TRIE_TEMPLATE_TYPE . . . 436
9.69.1.3 REPORT_COLLISION_WARNING . . . 436

9.70 inc/server/lm/models/h2d_map_trie.hpp File Reference . . . 437
9.71 inc/server/lm/models/layered_trie_base.hpp File Reference . . . 437
9.71.1 Macro Definition Documentation . . . 438
9.71.1.1 INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE . . . 438
9.71.1.2 LAYERED_BASE_ENSURE_CONTEXT . . . 438
9.72 inc/server/lm/models/m_gram_query.hpp File Reference . . . 439
9.73 inc/server/lm/models/w2c_array_trie.hpp File Reference . . . 439
9.74 inc/server/lm/models/w2c_hybrid_trie.hpp File Reference . . . 440
9.75 inc/server/lm/models/w2ch_um_storage.hpp File Reference . . . 441
9.76 inc/server/lm/models/word_index_trie_base.hpp File Reference . . . 441
9.77 inc/server/lm/proxy/lm_fast_query_proxy.hpp File Reference . . . 442
9.78 inc/server/lm/proxy/lm_fast_query_proxy_local.hpp File Reference . . . 442
9.79 inc/server/lm/proxy/lm_proxy.hpp File Reference . . . 443
9.80 inc/server/lm/proxy/lm_proxy_local.hpp File Reference . . . 443
9.81 inc/server/lm/proxy/lm_slow_query_proxy.hpp File Reference . . . 444
9.82 inc/server/lm/proxy/lm_slow_query_proxy_local.hpp File Reference . . . 444
9.83 inc/server/rm/builders/rm_basic_builder.hpp File Reference . . . 445
9.84 inc/server/rm/models/rm_basic_model.hpp File Reference . . . 445
9.85 inc/server/rm/models/rm_entry.hpp File Reference . . . 446
9.86 inc/server/rm/models/rm_query.hpp File Reference . . . 446
9.87 inc/server/rm/proxy/rm_proxy.hpp File Reference . . . 447
9.88 inc/server/rm/proxy/rm_proxy_local.hpp File Reference . . . 447
9.89 inc/server/rm/proxy/rm_query_proxy.hpp File Reference . . . 448
9.90 inc/server/rm/proxy/rm_query_proxy_local.hpp File Reference . . . 448
9.91 inc/server/rm/rm_configs.hpp File Reference . . . 448
9.92 inc/server/rm/rm_configurator.hpp File Reference . . . 449
9.93 inc/server/rm/rm_consts.hpp File Reference . . . 449
9.94 inc/server/rm/rm_parameters.hpp File Reference . . . 450
9.95 inc/server/server_configs.hpp File Reference . . . 450
9.95.1 Macro Definition Documentation . . . 450
9.95.1.1 SERVER_CONFIGS_HPP . . . 450
9.96 inc/server/server_consts.hpp File Reference . . . 451
9.97 inc/server/server_parameters.hpp File Reference . . . 451
9.98 inc/server/tm/builders/tm_basic_builder.hpp File Reference . . . 452
9.99 inc/server/tm/models/tm_basic_model.hpp File Reference . . . 452
9.100 inc/server/tm/models/tm_query.hpp File Reference . . . 453
9.101 inc/server/tm/models/tm_source_entry.hpp File Reference . . . 453
9.102 inc/server/tm/models/tm_target_entry.hpp File Reference . . . 454
9.103 inc/server/tm/proxy/tm_proxy.hpp File Reference . . . 454
9.104 inc/server/tm/proxy/tm_proxy_local.hpp File Reference . . . 455

9.105inc/server/tm/proxy/tm_query_proxy.hpp File Reference . . . 455
9.106inc/server/tm/proxy/tm_query_proxy_local.hpp File Reference . . . 456
9.107inc/server/tm/tm_configs.hpp File Reference . . . 456
9.108inc/server/tm/tm_configurator.hpp File Reference . . . 457
9.109inc/server/tm/tm_consts.hpp File Reference . . . 457
9.110inc/server/tm/tm_parameters.hpp File Reference . . . 457
9.111inc/server/trans_job_pool.hpp File Reference . . . 458
9.111.1 Macro Definition Documentation . . . 458
9.111.1.1 TRANS_JOB_POOL_HPP . . . 458
9.112inc/server/trans_task.hpp File Reference . . . 459
9.113inc/server/trans_task_id.hpp File Reference . . . 459
9.114inc/server/trans_task_pool.hpp File Reference . . . 459
9.115inc/server/trans_task_pool_worker.hpp File Reference . . . 460
9.116inc/server/translation_server.hpp File Reference . . . 460
9.116.1 Macro Definition Documentation . . . 461
9.116.1.1 ASIO_STANDALONE . . . 461
9.117README.md File Reference . . . 461
9.118src/client/bpbd_client.cpp File Reference . . . 461
9.118.1 Macro Definition Documentation . . . 461
9.118.1.1 PROGRAM_VERSION_STR . . . 461
9.118.2 Function Documentation . . . 461
9.118.2.1 create_arguments_parser() . . . 461
9.118.2.2 destroy_arguments_parser() . . . 461
9.118.2.3 main(int argc, char **argv) . . . 462
9.119src/client/trans_job_status.cpp File Reference . . . 462
9.119.1 Macro Definition Documentation . . . 462
9.119.1.1 STATUS_REQ_INITIALIZED_STR . . . 462
9.119.1.2 STATUS_REQ_SENT_FAIL_STR . . . 462
9.119.1.3 STATUS_REQ_SENT_GOOD_STR . . . 462
9.119.1.4 STATUS_RES_RECEIVED_STR . . . 462
9.119.1.5 STATUS_UNDEFINED_STR . . . 463
9.119.1.6 STATUS_UNKNOWN_STR . . . 463
9.120src/common/messaging/trans_job_code.cpp File Reference . . . 463
9.120.1 Macro Definition Documentation . . . 463
9.120.1.1 RESULT_CANCELED_STR . . . 463
9.120.1.2 RESULT_ERROR_STR . . . 463
9.120.1.3 RESULT_OK_STR . . . 463
9.120.1.4 RESULT_PARTIAL_STR . . . 464
9.120.1.5 RESULT_UNDEFINED_STR . . . 464
9.120.1.6 RESULT_UNKNOWN_STR . . . 464

9.121 src/common/utils/logging/logger.cpp File Reference . . . 464
9.121.1 Macro Definition Documentation . . . 464
9.121.1.1 IS_ENOUGH_LOGGING_LEVEL . . . 464
9.122 src/common/utils/monitor/statistics_monitor.cpp File Reference . . . 465
9.123 src/server/bpbd_server.cpp File Reference . . . 465
9.123.1 Macro Definition Documentation . . . 465
9.123.1.1 PROGRAM_VERSION_STR . . . 465
9.123.2 Function Documentation . . . 466
9.123.2.1 connect_to_models(const server_parameters &params) . . . 466
9.123.2.2 create_arguments_parser() . . . 466
9.123.2.3 destroy_arguments_parser() . . . 466
9.123.2.4 disconnect_from_models() . . . 466
9.123.2.5 main(int argc, char **argv) . . . 466
9.124 src/server/decoder/de_configurator.cpp File Reference . . . 466
9.125 src/server/lm/builders/lm_basic_builder.cpp File Reference . . . 466
9.125.1 Macro Definition Documentation . . . 467
9.125.1.1 INSTANTIATE_TRIE_BUILDER_FILE_READER . . . 467
9.126 src/server/lm/builders/lm_gram_builder.cpp File Reference . . . 468
9.126.1 Macro Definition Documentation . . . 468
9.126.1.1 INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL . . . 468
9.126.1.2 INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL_WEIGHT . . . 469
9.127 src/server/lm/lm_configurator.cpp File Reference . . . 469
9.128 src/server/lm/lm_query.cpp File Reference . . . 469
9.128.1 Macro Definition Documentation . . . 470
9.128.1.1 PROGRAM_VERSION_STR . . . 470
9.128.2 Function Documentation . . . 470
9.128.2.1 create_arguments_parser() . . . 470
9.128.2.2 destroy_arguments_parser() . . . 470
9.128.2.3 main(int argc, char **argv) . . . 470
9.129 src/server/lm/mgrams/byte_m_gram_id.cpp File Reference . . . 470
9.129.1 Macro Definition Documentation . . . 471
9.129.1.1 MAX_VALUE_IN_BYTES . . . 471
9.130 src/server/lm/mgrams/model_m_gram.cpp File Reference . . . 471
9.131 src/server/lm/mgrams/query_m_gram.cpp File Reference . . . 471
9.132 src/server/lm/models/c2d_hybrid_trie.cpp File Reference . . . 472
9.133 src/server/lm/models/c2d_map_trie.cpp File Reference . . . 472
9.134 src/server/lm/models/c2w_array_trie.cpp File Reference . . . 473
9.135 src/server/lm/models/g2d_map_trie.cpp File Reference . . . 473
9.136 src/server/lm/models/h2d_map_trie.cpp File Reference . . . 474
9.137 src/server/lm/models/m_gram_query.cpp File Reference . . . 475

9.138src/server/lm/models/w2c_array_trie.cpp File Reference 475
9.139src/server/lm/models/w2c_hybrid_trie.cpp File Reference 476
9.140src/server/rm/rm_configurator.cpp File Reference 476
9.141src/server/tm/models/tm_target_entry.cpp File Reference 477
9.142src/server/tm/tm_configurator.cpp File Reference 477
9.143src/server/trans_task_pool.cpp File Reference 477
9.144src/server/trans_task_pool_worker.cpp File Reference 478

Index **479**

Chapter 1

README

#**The Basic Phrase-Based Statistical Machine Translation Tool**

Author: Dr. Ivan S. Zapreev

Project pages: Git-Hub-Project

Introduction

This fork from the Back Off Language Model(s) for SMT project is aimed at creating an entire phrase-based statistical machine translation system. The delivered software follows a client/server architecture based on Web Sockets for C++ and consists of the three main applications::

- **bpbd-client** - a thin client to send the translation job requests to the translation server and obtain results
- **bpbd-server** - the translation server consisting of the following main components:
  - *Decoder* - the decoder component responsible for translating text from one language into another
  - *LM* - the language model implementation allowing for seven different trie implementations and responsible for estimating the target language phrase probabilities
  - *TM* - the translation model implementation required for providing source to target language phrase translation and the probabilities thereof
  - *RM* - the reordering model implementation required for providing the possible translation order changes and the probabilities thereof
- **lm-query** - a stand-alone language model query tool that allows to perform language model queries and estimate the joint phrase probabilities

To keep a clear view of the used terminology further, we provide some details on the topic of phrase-based SMT, and illustrate it the picture below, taken from TAUS MT SHOWCASE slides.

The entire phrase-based statistical machine translation relies on learning statistical correlations between words and phrases in an existing source/target translation text pair, also called parallel corpus or corpora. These correlations are learned by, generally speaking, three statistical models: TM - translation model; RM - reordering mode; and LM - language model; briefly mentioned above. If the training corpora is large enough, then these models will possess enough information to approximate a translation of an arbitrary text from the given source language to the given target language. Note that, before this information can be extracted, the parallel corpora undergoes the process called *word alignment* which is aimed at estimating which words/phrases in the source language correspond to which words/phrases in the target language. Let us give a more precise definition of these models:

1. Translation model - provides phrases in the source language with learned possible target language translations and the probabilities thereof.

2. Reordering model - stores information about probable translation orders of the phrases within the source text, based on the observed source and target phrases and their alignments.
3. Language model - reflects the likelihood of this or that phrase in the target language to occur. In other words, it is used to evaluate the obtained translation for being _"sound"_ in the target language.

Note that, the language model is typically learned from a different corpus in a target language.

With these three models at hand one can perform decoding, which is a synonym to a translation process. S↩ MT decoding is performed by exploring the state space of all possible translations and reordering of the source language phrases within one sentence. The purpose of decoding, as indicated by the maximization procedure at the bottom of the figure above, is to find a translation with the largest possible probability.

The rest of the document is organized as follows:

1. `Project structure` - Gives the file and folder structure of the project
2. `Supported platforms` - Indicates the project supported platforms
3. `Building the project` - Describes the process of building the project
4. `Using software` - Explain how the software is to be used
5. `Input file formats` - Provides examples of the input file formats
6. `Code documentation` - Refers to the project documentation
7. `External libraries` - Lists the included external libraries
8. `Performance evaluation` - Contains performance evaluation results
9. `General design` - Outlines the general software design
10. `Software details` - Goes about some of the software details
11. `Literature and references` - Presents the list of used literature
12. `Licensing` - States the licensing strategy of the project
13. `History` - Stores a short history of this document

Project structure

This is a Netbeans 8.0.2 project, based on cmake, and its top-level structure is as follows:

- `**[Project-Folder]**/`
  - **doc**/ - contains the project-related documents including the Doxygen-generated code documentation and images
  - **ext**/ - stores the external header only libraries used in the project
  - **inc**/ - stores the C++ header files of the implementation
  - **src**/ - stores the C++ source files of the implementation
  - **nbproject**/ - stores the Netbeans project data, such as makefiles
  - **data**/ - stores the test-related data such as test models and query input files, as well as some experimental results
  - default.cfg - an example server configuration file
  - LICENSE - the code license (GPL 2.0)
  - CMakeLists.txt - the cmake build script for generating the project's make files
  - README.md - this document
  - Doxyfile - the Doxygen configuration file

Supported platforms

This project supports two major platforms: Linux and Mac Os X. It has been successfully build and tested on:

- **Centos 6.6 64-bit** - Complete functionality.
- **Ubuntu 15.04 64-bit** - Complete functionality.
- **Mac OS X Yosemite 10.10 64-bit** - Limited by inability to collect memory-usage statistics.

Notes:

1. There was only a limited testing performed on 32-bit systems.
2. The project must be possible to build on Windows platform under `Cygwin`.

Building the project

Building this project requires **gcc** version >= *4.9.1* and **cmake** version >= 2.8.12.2.

The first two steps before building the project, to be performed from the Linux command line console, are:

1. `cd [Project-Folder]`
2. `mkdir build`

After these are performed, the project can be build in two ways:

- From the Netbeans environment by running Build in the IDE
  - In Netbeans menu: `Tools/Options/"C/C++"` make sure that the cmake executable is properly set.
  - Netbeans will always run cmake for the DEBUG version of the project
  - To build project in RELEASE version use building from Linux console
- From the Linux command-line console by following the next steps
  - `cd [Project-Folder]/build`
  - `cmake -DCMAKE_BUILD_TYPE=Release ..` OR `cmake -DCMAKE_BUILD_TYP↩E=Debug ..`
  - `make -j [NUMBER-OF-THREADS]` add `VERBOSE=1` to make the compile-time options visible

The binaries will be generated and placed into ∗./build/∗ folder. In order to clean the project from the command line run `make clean`. Cleaning from Netbeans is as simple calling the `Clean and Build` from the `Run` menu.

Project compile-time parameters

For the sake of performance optimizations, the project has a number of compile-time parameters that are to be set before the project is build and can not be modified in the runtime. Let us consider the most important of them and indicate where all of them are to be found.

Logging level: Logging is important when debugging software or providing an additional user information during the program's runtime. Yet additional output actions come at a price and can negatively influence the program's performance. This is why it is important to be able to disable certain logging levels within the program not only during its runtime but also at compile time. The possible range of project's logging levels, listed incrementally, is: ERROR, WARNING, USAGE, RESULT, INFO, INFO1, INFO2, INFO3, DEBUG, DEBUG1, DEBUG2, DEBUG3,

DEBUG4. One can limit the logging level range available at runtime by setting the LOGER_M_GRAM_LEVEL_↩ MAX constant value in the ./inc/common/utils/logging/logger.hpp header file. The default value is INFO3.

Sanity checks: When program is not running as expected, it could be caused by the internal software errors that are potentially detectable at runtime. This software has a number of build-in sanity checks that can be enabled/disabled at compile time by setting the DO_SANITY_CHECKS boolean flag in the ./inc/common/utils/exceptions.hpp header file. Note that enabling the sanity checks does not guarantee that the internal error will be found but will have a negative effect on the program's performance. Yet, it might help to identify some of the errors with e.g. input file formats and alike.

Server configs: There is a number of translation server common parameters used in decoding, translation, re-ordering and language models. Those are to be found in the ./inc/server/server_configs.hpp:

- UNKNOWN_LOG_PROB_WEIGHT - The value used for the unknown probability weight _(log10 scale)_
- ZERO_LOG_PROB_WEIGHT - The value used for the 'zero' probability weight _(log10 scale)_
- tm::NUM_TM_FEATURES - The number of the translation model features, which defines the exact number of features read per entry from the translation model input file
- tm::TM_MAX_TARGET_PHRASE_LEN - The maximum length of the target phrase to be considered, this defines the maximum number of tokens to be stored per translation entry
- lm::NUM_LM_FEATURES - The number of language model features, the program currently supports only one value: 1
- lm::LM_M_GRAM_LEVEL_MAX - The language model maximum level, the maximum number of words in the language model phrase
- lm::LM_HISTORY_LEN_MAX - **do not change** this parameter
- lm::LM_MAX_QUERY_LEN - **do not change** this parameter
- lm::DEF_UNK_WORD_LOG_PROB_WEIGHT - The default unknown word probability weight, for the case the <unk> entry is not present in the language model file _(log10 scale)_
- rm::NUM_RM_FEATURES - The number of reordering model features, the only two currently supported values are: 6 and 8

Decoder configs: The decoder-specific parameters are located in ./inc/server/decoder/de_↩ configs.hpp:

- MAX_WORDS_PER_SENTENCE - The maximum allowed number of words/tokens per sentence to translate.

LM configs: The Language-model-specific parameters located in ./inc/server/lm/lm_configs.hpp:

- lm_word_index - the word index type to be used, the possible values are:
  - basic_word_index - the basic word index that just loads the uni-grams in the same order as in the LM model file and gives them consecutive id values.
  - counting_word_index - the basic word index that counts the number of times the uni-gram occurs in the LM model file and gives lower ids to the more frequent uni-grams. This ensures some performance boost (within 10%) when querying certain types of language models but requires longer loading times.
  - optimizing_word_index<basic_word_index> - the optimizing word index is based on the linear probing hash map so it is the fastest, it uses a basic word index as a bootstrap word index for issuing the ids.
  - optimizing_word_index<counting_word_index> - the optimizing word index is based on the linear probing hash map so it is the fastest, it uses a counting word index as a bootstrap word index for issuing the ids.

  - hashing_word_index - the hashing word index is a discontinuous word index that does not issue the uni-gram ids consequently but rather associates each uni-gram with its hash value, the latter is taken to be a unique identifier. This is the only type of index supported by the hash-based h2d_map_trie.

- lm_model_type - the trie model type to be used, the possible values (trie types) are as follows, for a performance comparison thereof see Performance Evaluation:
  - c2d_hybrid_trie<lm_word_index> - contains the context-to-data mapping trie implementation based on std::unordered map and ordered arrays
  - c2d_map_trie<lm_word_index> - contains the context-to-data mapping trie implementation based on std::unordered map
  - c2w_array_trie<lm_word_index> - contains the context-to-word mapping trie implementation based on ordered arrays
  - g2d_map_trie<lm_word_index> - contains the m-gram-to-data mapping trie implementation based on self-made hash maps
  - h2d_map_trie<lm_word_index> - contains the hash-to-data mapping trie based on the linear probing hash map implementation
  - w2c_array_trie<lm_word_index> - contains the word-to-context mapping trie implementation based on ordered arrays
  - w2c_hybrid_trie<lm_word_index> - contains the word-to-context mapping trie implementation based on std::unordered map and ordered arrays

- lm_model_reader - the model reader is basically the file reader type one can use to load the model, currently there are three model reader types available, with cstyle_file_reader being the default:
  - file_stream_reader - uses the C++ streams to read from files, the slowest
  - cstyle_file_reader - uses C-style file reading functions, faster than file_stream_reader
  - memory_mapped_file_reader - uses memory-mapped files which are faster than the cstyle_file_reader but consume twice the file size memory (virtual RAM).

- lm_builder_type - currently there is just one builder type available: lm_basic_builder<lm_↩ model_reader>.

Note that not all of the combinations of the lm_word_index and lm_model_type can work together, this is reported runtime after the program is build. Some additional details on the preferred configurations can be also found in the ./inc/server/lm/lm_consts.hpp header file comments. The default, and the most optimal performance/memory ratio configuration, is:

- lm_word_index being set to hashing_word_index
- lm_model_type begin set to h2d_map_trie<lm_word_index>.

TM configs: The Translation-model-specific parameters are located in ./inc/server/tm/tm_↩ configs.hpp:

- tm_model_type - currently there is just one model type available: tm_basic_model
- tm_model_reader - the same as lm_model_reader for _"LM configs"_, see above
- tm_builder_type - currently there is just one builder type available: tm_basic_builder<tm_↩ model_reader>

RM configs: The Reordering-model-specific parameters are located in ./inc/server/rm/rm_↩ configs.hpp:

- rm_model_type - currently there is just one model type available: rm_basic_model
- rm_model_reader - the same as lm_model_reader for _"LM configs"_, see above
- rm_builder_type - currently there is just one builder type available: rm_basic_builder<rm_↩ model_reader>

Using software

This section briefly covers how the provided software can be used for performing text translations. We begin with the **bpbd-server** and the **bpbd-client** then briefly talk about the **lm-query**. For information on the LM, TM and RM model file formats and others see section `Input file formats`

Translation server: *bpbd-server*

The translation server is used for two things: _(i)_ to load language, translation and reordering models (for a given source/target language pair); _(ii)_ to process the translation requests coming from the translation client. The use of this executable is straightforward. When started from a command line without any parameters, **bpbd-server** reports on the available command-line options:

```
1 $ bpbd-server
2 <...>
3 PARSE ERROR:
4              Required argument missing: config
5
6 Brief USAGE:
7    bpbd-server  [-d <error|warn|usage|result|info|info1|info2|info3>] -c
8                 <server configuration file> [--] [--version] [-h]
9
10 For complete USAGE and HELP type:
11    bpbd-server --help
```

As one can see the only required command-line parameter of the translation server is a configuration file. The latter shall contain the necessary information for loading the models, and running the server. The configuration file content is covered in section `Configuration file` below. Once the translation server is started there is still a way to change some of its run-time parameters. The latter can be done with a server console explained in the `Server console` section below. In addition, for information on the LM, TM and RM model file formats see the `Input file formats`

Configuration file

In order to start the server one must have a valid configuration file for it. The latter stores the minimum set of parameter values needed to run the translation server. Among other things, this config file specifies the location of the language, translation and reordering models, the number of translation threads, and the web socket port through which the server will accept requests. An example configuration file can be found in: `[Project-Folder]/default.cfg` and in `[Project-Folder]/data`. The content of this file is self explanatory and contains a significant amount of comments.

When run with a properly formed configuration file, **bpbd-server** gives the following output. Note the `-d info1` option ensuring additional information output during loading the models.

```
1 $ bpbd-server -c ../data/default-1-3.000.000.cfg -d info1
2 <...>
3 USAGE: The requested debug level is: 'INFO1', the maximum build level is 'INFO3' the set level is 'INFO1'
4 USAGE: Loading the server configuration option from: ../data/default-1-3.000.000.cfg
5 USAGE: Translation server from 'German' into 'English' on port: '9002' translation threads: '25'
6 INFO: LM parameters: [ conn_string = ../data/models/e_30_2564372.lm, num_lm_feature_weights = 1,
       lm_feature_weights = [ 1 ] ]
7 INFO: TM parameters: [ conn_string = ../data/models/de-en-1-3.000.000.tm, num_tm_feature_weights = 4,
       tm_feature_weights = [ 1|1|1|1 ], translation_limit = 30, min_trans_prob = 1e-20 ]
8 INFO: RM parameters: [ conn_string = ../data/models/de-en-1-3.000.000.rm, num_rm_feature_weights = 6,
       rm_feature_weights = [ 1|1|1|1|1|1 ] ]
9 INFO: DE parameters: [ distortion = 5, ext_dist_left = 1, num_best_trans = 10, pruning_threshold = 1.1,
       stack_capacity = 100, word_penalty = -0.3, phrase_penalty = 1.2, max_source_phrase_len = 7,
       max_target_phrase_len = 7 ]
10 USAGE: ----------------------------------------------------------
11 USAGE: Start creating and loading the Language Model ...
12 USAGE: Language Model is located in: ../data/models/e_30_2564372.lm
13 USAGE: Using the <cstyle_file_reader.hpp> file reader!
14 USAGE: Using the <h2d_map_trie.hpp> model.
15 INFO: The <h2d_map_trie.hpp> model's buckets factor: 2
16 INFO: Expected number of M-grams per level: [ 199164 4202658 15300577 26097321 31952150 ]
17 INFO1: Pre-allocating memory:  0 hour(s) 0 minute(s) 0 second(s)
18 INFO1: Reading ARPA 1-Grams:  0 hour(s) 0 minute(s) 0 second(s)
19 INFO1: Reading ARPA 2-Grams:  0 hour(s) 0 minute(s) 5 second(s)
20 INFO1: Reading ARPA 3-Grams:  0 hour(s) 0 minute(s) 27 second(s)
21 INFO1: Reading ARPA 4-Grams:  0 hour(s) 0 minute(s) 56 second(s)
```

```
22 INFO1: Reading ARPA 5-Grams:  0 hour(s) 1 minute(s) 16 second(s)
23 USAGE: Reading the Language Model took 170.276 CPU seconds.
24 USAGE: Action: 'Loading the Language Model' memory change:
25 USAGE: vmsize=+1770 Mb, vmpeak=+1770 Mb, vmrss=+1771 Mb, vmhwm=+1771 Mb
26 USAGE: ------------------------------------------------------
27 USAGE: Start creating and loading the Translation Model ...
28 USAGE: Translation Model is located in: ../data/models/de-en-1-3.000.000.tm
29 USAGE: Using the <cstyle_file_reader.hpp> file reader!
30 USAGE: Using the hash-based translation model: tm_basic_model.hpp
31 INFO1: Counting phrase translations:  0 hour(s) 0 minute(s) 10 second(s)
32 INFO: The number of valid TM source entries is: 1620524
33 INFO1: Building translation model:  0 hour(s) 0 minute(s) 43 second(s)
34 USAGE: Reading the Translation Model took 58.8196 CPU seconds.
35 USAGE: Action: 'Loading the Translation Model' memory change:
36 USAGE: vmsize=+550 Mb, vmpeak=+550 Mb, vmrss=+550 Mb, vmhwm=+550 Mb
37 USAGE: ------------------------------------------------------
38 USAGE: Start creating and loading the Reordering Model ...
39 USAGE: Reordering Model is located in: ../data/models/de-en-1-3.000.000.rm
40 USAGE: Using the <cstyle_file_reader.hpp> file reader!
41 USAGE: Using the hash-based reordering model: rm_basic_model.hpp
42 INFO1: Counting reordering entries:  0 hour(s) 0 minute(s) 6 second(s)
43 INFO: The number of RM source/target entries matching TM is: 2567397
44 INFO1: Building reordering model:  0 hour(s) 0 minute(s) 12 second(s)
45 USAGE: Reading the Reordering Model took 21.6754 CPU seconds.
46 USAGE: Action: 'Loading the Reordering Model' memory change:
47 USAGE: vmsize=+78 Mb, vmpeak=+61 Mb, vmrss=+78 Mb, vmhwm=+61 Mb
48 USAGE: The server is started!
49 <...>
```

In the first seven lines we see information loaded from the configuration file. Further, the LM, TM, and RM, models are loaded and the information thereof is provided. Note that for less output one can simply run `bpbd-server -c ../data/default-1-3.000.000.cfg`.

There is a few important things to note about the configuration file at the moment:

- `[Translation Models]/tm_feature_weights` - the number of features must be equal to the value of `tm::NUM_TM_FEATURES`, see Project compile-time parameters.
- `[Translation Models]/tm_unk_features` - the number of features must be equal to the value of `tm::NUM_TM_FEATURES`, see Project compile-time parameters.
- `[Reordering Models]/rm_feature_weights` - the number of features must be equal to the value of `lm::NUM_RM_FEATURES`, see Project compile-time parameters.
- `[Language Models]/lm_feature_weights` - the number of features must be equal to the value of `lm::NUM_LM_FEATURES`, see Project compile-time parameters.

Server console

Once the server is started it is not run as a Linux daemon but is a simple multi-threaded application that has its own interactive console allowing to manage some of the configuration file parameters and obtain some run-time information about the server. The list of available server console commands is given in the listing below:

```
1 $ bpbd-server -c ../data/default-1-3.000.000.cfg -d info2
2 <...>
3 USAGE: The server is started!
4 USAGE: Available server commands:
5 USAGE:  'q & <enter>'  - to exit.
6 USAGE:  'h & <enter>'  - print HELP info.
7 USAGE:  'r & <enter>'  - run-time statistics.
8 USAGE:  'p & <enter>'  - print server parameters.
9 USAGE:  'set ll <level> & <enter>'  - set log level.
10 USAGE:  'set nt  <positive integer> & <enter>'  - set the number of worker threads.
11 USAGE:  'set nbt <unsigned integer> & <enter>'  - set the number of best translations.
12 USAGE:  'set d <integer> & <enter>'  - set the distortion limit.
13 USAGE:  'set edl <unsigned integer> & <enter>'  - set the extra left distortion.
14 USAGE:  'set pt <unsigned float> & <enter>'  - set pruning threshold.
15 USAGE:  'set sc <integer> & <enter>'  - set stack capacity.
16 USAGE:  'set wp <float> & <enter>'  - set word penalty.
17 USAGE:  'set pp <float> & <enter>'  - set phrase penalty.
18 >>
```

Note that, the commands allowing to change the translation process, e.g. the stack capacity, are to be used with great care. For the sake of memory optimization, **bpbd-server** has just one copy of the server runtime parameters

used from all the translation processes. So in case of active translation process, changing these parameters can cause disruptions thereof starting from an inability to perform translation and ending with memory leaks. All newly scheduled or finished translation tasks however will not experience any disruptions.

Translation client: *bpbd-client*

The translation client is used to communicate with the server by sending translation job requests and receiving the translation results. When started from a command line without any parameters, **bpbd-client** reports on the available command-line options:

```
1 $bpbd-client
2 <...>
3 PARSE ERROR:
4              Required arguments missing: output-file, input-lang, input-file
5
6 Brief USAGE:
7    bpbd-client  [-d <error|warn|usage|result|info|info1|info2|info3>] [-t]
8                 [-l <min #sentences per request>] [-u <max #sentences per
9                 request>] [-p <server port>] [-s <server address>] [-o
10                 <target language>] -O <target file name> -i <source
11                 language> -I <source file name> [--] [--version] [-h]
12
13 For complete USAGE and HELP type:
14    bpbd-client --help
```

One of the main required parameters of the translation client is the input file. The latter should contain text in the source language to be translated into the target one. The input file is expected to have one source language sentence per line. The client application does have a basic algorithm for tokenising sentences and putting them into the lower case, i.e. preparing each individual sentence for translation but this algorithm is pretty rudimental. Therefore, it is suggested that the input file should not only contain one sentence per line but each sentence must be provided in a tokenized (space-separated), lower-case format.

Once started, the translation client makes a web socket connection to the translation server, reads text from the input file, splits it into a number of translation job requests (which are sent to the translation server) and waits for the reply. Each translation job sent to the server consists of a number of sentences called translation tasks. The maximum and minimum number of translation tasks per a translation job is configurable via additional client parameters. For more info run: `bpbd-client --help`.

Once the translations are performed, and the translation job responses are received, the resulting text is written to the output file. Each translated sentence is put on a separate line in the same order it was seen in the input file. Each output line/sentence also gets prefixed with a translation status having a form: `<status>`. If a translation task was cancelled, or an error has occurred then it is indicated by the status and the information about that is also placed in the output file on the corresponding sentence line.

Remember that, running **bpbd-client** with higher logging levels will give more insight into the translation process and functioning of the client. It is also important to note that, the source-language text in the input file is must be provided in the **UTF8** encoding.

Language model query tool: *lm-query*

The language model query tool is used for querying stand alone language models to obtain the joint m-gram probabilities. When started from a command line without any parameters, **lm-query** reports on the available command-line options:

```
1 $ lm-query
2 <...>
3 PARSE ERROR:
4              Required arguments missing: query, model
5
6 Brief USAGE:
7    lm-query  [-l <lm lambda weight>] [-d <error|warn|usage|result|info
8              |info1|info2|info3>] -q <query file name> -m <model file name>
9              [--] [--version] [-h]
10
11 For complete USAGE and HELP type:
12    lm-query --help
```

For information on the LM file format see section Input file formats. The query file format is a text file in a **UTF8** encoding which, per line, stores one query being a space-separated sequence of tokens in the target language. The maximum allowed query length is limited by the compile-time constant `lm::LM\_MAX\_QUER↩ Y\_LEN`, see section Project compile-time parameters

Input file formats

In this section we briefly discuss the model file formats supported by the tools. We shall occasionally reference the other tools supporting the same file formats and external third-party web pages with extended format descriptions.

Translation model: `*.tm`

The translation-model file stores the phrase pairs in the source and target languages and the pre-computed probability weights in the following strict format:

```
1 <source-phrase> ||| <target-phrase> ||| <prob-1> <prob-2> <prob-3> <prob-4>
```

As generated by, e.g. Moses. In general the source and target phrases and target phrase and probability weight sections are separated by five symbols: one space three vertical lines and one space. Source and target space words must be space separated, as well as the probability weights. At the moment, everything followed after the fourth probability, until the end of the line, is ignored. The tool supports `4` translation probabilities and the supported number of weights is defined by the `tm::NUM_TM_FEATURES` constant value, see Project compile-time parameters. If the format is not followed, the program's behavior is not specified.

Reordering model: `*.rm`

The reordering-model file stores the phrase pairs in the source and target languages and the reordering weights in the following strict format:

```
1 <source-phrase> ||| <target-phrase> ||| <weight-1> <weight-2> ... <weight-k>
```

As generated by, e.g. Moses. In general the source and target phrases and target phrase and probability weight sections are separated by five symbols: one space three vertical lines and one space. Source and target space words must be space separated, as well as the probability weights. At the moment, everything followed after the last probability, until the end of the line, is ignored. The number weights `k` is fixed per model file. The tool supports `6` or `8` reordering weights and the supported number of weights is defined by the `rm::NUM_RM_FEATURES` constant value, see Project compile-time parameters. If the format is not followed, the program's behavior is not specified.

Language model: `*.lm`

The language model file is a UTF8 text file in a well known ARPA format, see e.g. details on MSDN help or Speech Technology and Research (STAR) Laboratory. An example ARPA file is given below:

```
1 <header - information ignored by applications>
2
3 \data\
4 ngram 1=9
5 ngram 2=11
6 ngram 3=3
7
8 \1-grams:
9 -0.8953 <unk>        -0.7373
10 -0.7404 </s> -0.6515
11 -0.7861 <s>   -0.1764
12 -1.0414 When -0.4754
13 -1.0414 will -0.1315
14 -0.9622 the   0.0080
15 -1.4393 Stock        -0.3100
16 -1.0414 Go    -0.3852
```

```
17 -0.9622 Up    -0.1286
18
19 \2-grams:
20 -0.3626 <s> When     0.1736
21 -1.2765 <s> the      0.0000
22 -1.2765 <s> Up       0.0000
23 -0.2359 When will    0.1011
24 -1.0212 will </s>    0.0000
25 -0.4191 will the     0.0000
26 -1.1004 the </s>     0.0000
27 -1.1004 the Go       0.0000
28 -0.6232 Stock Go     0.0000
29 -0.2359 Go Up        0.0587
30 -0.4983 Up </s>
31
32 \3-grams:
33 -0.4260 <s> When will
34 -0.6601 When will the
35 -0.6601 Go Up </s>
36
37 \end\
```

Note that the format is expected to be followed in a very strict way. The headers can be skipped, the empty lines must be empty, the M-gram entry:

```
1 <probability>    <word-1> <word-2> ... <word-m>    <back-off-weight>
```

Must have one *tabulation* symbol after the `<probability>`, single space between any two words, and a single *tabulation* symbol before the `<back-off-weight>`. If the format is not followed, the program's behavior is not specified. The maximum allowed language model level, the maximum value of N in the N-gram, is defined by the compile-time parameter `lm::LM_M_GRAM_LEVEL_MAX`, see Project compile-time parameters.

Code documentation

At present the documentation is done in the Java-Doc style that is successfully accepted by Doxygen with the Doxygen option *JAVADOC_AUTOBRIEF* set to *YES*. The generated documentation is located in two folders:

- `[Project-Folder]/doc/html`
  - Open the *index.html* file located in this folder with your favorite web browser.
- `[Project-Folder]/doc/latex`
  - Open the *refman.pdf* file located in this folder with your favorite pdf viewer.

The `[Project-Folder]/Doxyfile` can be used to re-generate the documentation at any given time, for more details see Doxygen.

- To re-build the Latex documentation run the following commands from the Linux console:
  - `cd [Project-Folder]/doc/latex`
  - `make`

External libraries

At present this project uses the following external/third-party header-only libraries:

Library Name	Purpose	Website	Version	Licence
Feather ini parser	_Fast, lightweight, header, portable INI/configuration file parser for ANSI C++._	link	1.40	MIT
WebSocket++	_Is an open source, header only C++ library implementing RFC6455 (The WebSocket Protocol)._	link	0.6.0	BSD
Asio C++ Library	_A cross-platform C++ library for network and low-level I/O programming_	link	1.10.6	Boost
Tclap	_A small and flexible library that provides a simple interface for defining and accessing command line arguments_	link	1.2.1	MIT

Performance evaluation

In this section we provide an empirical comparison of the developed LM query tool with two other well known tools, namely SRILM and KenLM, both of which provide language model implementations that can be queried. The additional information on the compared tools is to be found in Appendix Tests

Test set-up

The main target of this experimental comparison is to evaluate memory consumption and query times of the implemented tries. For doing that we do not rely on the time and memory statis- tics reported by the tools but rather, for the sake of uniform and independent opinion, rely on the Linux standard time utility available in the `zsh` Linux shell. The latter provides system- measured statistics about the program run. We choose to measure:

- **MRSS** - the maximum resident memory usage of the program
- **CPU time** - the CPU time in seconds

We chose to measure maximum resident memory usage as this is what defines the amount of RAM needed to run the program. Also, the CPU times are the actual times that the program was executed on the CPU. Measuring CPU times allows for a fair comparison as excludes possible results influence by the other system processes.

The experiments were set up to be run with different-size 5-gram language models given in the ARPA format with two types of inputs:

1. The single 5-gram query that defines the baseline
2. The file input with 100,000,000 of 5-gram queries

The delta in execution CPU times between the baseline and the 100,000,000 query files defines the pure query execution time of the tool. Note that, the query files were produced from the text corpus different from the one used

to produce the considered language models. The MRSS values are reported in gigabytes (Gb) and the CPU times are measured in seconds. The plots provide MRSS and CPU times relative to the input model size in Gb.

The test hardware configuration and the model/query files' data is to be found in Appendix Tests

Experimental results

The experimental results are present in the following two pictures. The first one indicates the changes in the MRSS depending on the model size:

The second one shows the query CPU times depending on the model sizes:

The results show that the developed LM model trie representations are highly compatible with the available state of the art tools. We also give the following usage guidelines for the implemented tries:

- **w2ca** and **c2wa** tries are beneficial for the machines with limited RAM. If low memory usage is very critical then bitmap hash caching can also be disabled.
- **c2dm** trie provides the fastest performance with moderate memory consumption. This is recommended when high performance is needed but one should be aware of possible m-gram id collisions.10
- **c2dh** trie is preferable if performance, as well as moderate memory consumption, is needed. This is the second-fastest trie which, unlike **c2dm**, is fully reliable.
- **w2ch** trie did not show itself useful and **g2dm** is yet to be re-worked and improved for better performance and memory usage.
- **h2dm** following the intuitions of the KenLM implementation, realizes the hash-map based trie using the linear probing hash map which turns to be the fastest trie with one of the best memory consumption. This tries type is used as a default one

General design

This section describes the ultimate and the current designs of the provided software. Note that the designs below are schematic only and the actual implementation might deviate. Yet, they are sufficient to reflect the overall structure of the software. We first provide the ultimate design we are going to work for and then give some insights into the currently implemented version thereof.

The designs were created using Unified Modeling Language (UML) with the help of the online UML tool called UMLetino.

The ultimate design

Consider the deployment diagram below. It shows the ultimate design we are aiming at.

This design's main feature is that it is fully distributed, and consists of three, vertical, layers.

1. *The first layer* - (located on the left side), is the front desk-load balancing piece of software who's responsibility is receiving the translation job requests from one language to another and then forwarding them to the second layer of the design, performing load balancing.
2. *The second layer* - (located in the middle), is a number of decoding servers that execute translation jobs. These servers can run decoders performing one-to-one language translations, and there may be multiple instances thereof for the same or different models. To generalize, each decoder might be able to translate from a bunch of languages into a bunch of other languages and then the middle layer servers can run one or more multiple instances of similarly or differently configured decoders, each.
3. *The third layer* - (located on the right side), is the layer of various instances of the Language, Translation, and Reordering models. Once again, each server can can run multiple instances of the same or different models to distribute the workload. Any decoder is free to use any number of model instances running in the third layer.

The communication between the layers here is suggested to be done using Web sockets as from industry it is known to be the fastest non-proprietary asynchronous communication protocol over TCP/IP. However, in case of significant network communication overhead the design allows for the system components to be run locally on the same physical computing unit or even to be build into a monolithic application for a complete avoidance of the socket communications. The latter is achieved by simply providing a local implementation of the needed system component. This approach is exactly an taken in the first version of the implemented software discussed in the next section.

The current design

Due to the limited time and as a proof of concept, the first version of the project follows the simplified version of the ultimate design given by the deployment diagram below.

As one can notice, in this figure the first layer is removed, i.e. there is no load-balancing entity. Also the Language, Translation, and Reordering models have local interface implementations only and are compiled together with the decoder component to form a single application. Of course, one can easily extend this design towards the ultimate one by simply providing the remove implementations for the LM, TM and RM models using the existing interfaces and implemented LM, RM and TM libraries.

Let us now briefly consider the two most complicated components of the software, the *Decoder* and the *Language model*.

The decoder component

The class diagram of the decoder component is given below. The decoder has a multi-threaded implementation, where each translation job (*a number of sentences to translate*) gets split into a number of translations tasks (*one task is one sentence*). Every translation task is executed in a separate thread. The number of translation threads is configurable at any moment of time.

Let us consider the main classes from the diagram:

The *translation_server* is responsible for: receiving the Web socket session open and close requests; parsing the translation requests into translation jobs; scheduling the translation jobs to the _trans_job_pool; receiving the finished job notification; and sending the finished job reply to the client.

The *trans_job_pool* stores all the scheduled translation jobs and splits them into the translation tasks scheduled by the *trans_task_pool*. Once all the translation tasks of a translation job are finished the *trans_job* notifies the *trans_job_pool and that, in its turn notifies the _translation_server*.

The *trans_task_pool* contains the queue of scheduled translation tasks and a limited number of translation worker threads to perform translations. In essence this is a thread pool entity with a queue of thread tasks.

The *trans_task* is a simple wrapper around the sentence translation entity *sentence_decoder*. The latter's responsibility is to retrieve the preliminary information from the Language, Translation, and Reordering models and then to perform translations using the *multi_tack* class, and instances of *stack_level* and *stack_state* classes. The latter represents the translation expansion hypothesis. At present the translation algorithm supports:

- Beam search
- Future cost estimates
- Threshold pruning of hypothesis
- Histogram pruning of hypothesis
- Hypothesis recombination

The LM component

Let us now consider the LM implementation class/package diagram on the figure below:

The design of the Language model has not changed much since the split off from the `Back Off Language Model SMT` project. So for more details we still refer to the `Implementation Details section` of the

README.md thereof. For the most recent information on the LM component design please read the project's Code documentation.

Software details

In this section we provide some additional details on the structure of the provided software. We shall begin with the common packages and then move on to the binary specific ones. The discussion will not go into details and will be kept at the level of source file folder, explaining their content.

Note that, to the possible extend the software is implemented via the header files located in the `[Project-Folder]/inc`. Due to the C++ language restrictions some of the header files do have corresponding C++ source files located in `[Project-Folder]/src`. The latter, to the necessary extend, follows the structure and file names found in `[Project-Folder]/inc`. Therefore, further we will only concentrate on the content of the `[Project-Folder]/inc` folder.

Additional information about the source code implementation classes can be found in the project's Code documentation.

common packages

The project's common packages are located in `[Project-Folder]/inc/common`:

- `/messaging` - web-socket message related classes common for the bpbd server and client
- `/utils` - various utility classes and libraries needed for logging and etc.
  - `/containers` - some container type classes
  - `/file` - file-reading related classes
  - `/logging` - logging classes
  - `/monitor` - memory usage and CPU times monitor classes

bpbd-client

All of the *bpbd-client* specific implementation classes are located in `[Project-Folder]/inc/client`.

bpbd-server

All of the *bpbd-server* specific implementation classes are located in `[Project-Folder]/inc/server`:

- `/common` - classes common to all server components
  - `/models` - model-related classes common to all server components
- `/decoder` - classes used in the decoder component
  - `sentence` - classes related to the top-level sentence decoding algorithms
  - `stack` - the multi-stack classes related to the stack-based decoding algorithms
- `/tm` - the translation model classes
  - `builders` - the model builder classes needed for reading the models
  - `models` - the model representation classes
  - `proxy` - the proxy objects implementing the local and/or remote model interface
- `/rm` - the reordering model classes
  - The same as for `/tm`.
- `/lm` - the language model classes
  - Similar to `/tm` and `/rm` but has some differences, see the next sub-section.

lm-query

All of the *lm-query* specific implementation classes are located in `[Project-Folder]/inc/server/lm/`. The structure of this folder follows the general patters of that of `[Project-Folder]/inc/server/tm/` and `[Project-Folder]/inc/server/rm/` but has the following additional sub-folders:

- `/dictionaries` - dictionary/word-index related classes
- `/mgrams` - model and query m-gram related classes.

Literature and references

This project is originally based on the following literature:

- Kenneth Heafield. "Kenlm: Faster and smaller language model queries." BibTex
- Philipp Koehn. "Statistical Machine Translation". BibTex
- Mark Jan Nederhof, Giorgio Satta. "Prefix Probability for Probabilistic Synchronous Context-Free Grammars" BibTex
- Adam Pauls, Dan Klein. "Faster and Smaller N-Gram Language Models" BibTex
- Daniel Robenek, Jan Platos, Vaclav Snasel. "Efficient In-memory Data Structures for n-grams Indexing" BibTex
- Andreas Stolcke, Jing Zheng, Wen Wang, Victor Abrash. "SRILM at Sixteen: Update and Outlook" BibTex
- Matthew Szudzik. "An Elegant Pairing Function" BibTex

Licensing

This is a free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This software is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see http://www.gnu.org/licenses/.

History

- **21.04.2015** - Created
- **27.07.2015** - Changed project name and some to-do's
- **21.09.2015** - Updated with the latest developments preparing for the version 1, Owl release.
- **11.03.2016** - Updated to reflect the latest project status.

Appendix Tests

SRILM

Is a toolkit for building and applying statistical language models (LMs), primarily for use in speech recognition, statistical tagging and segmentation, and machine translation. It has been under development in the SRI Speech Technology and Research Laboratory since 1995. The employed tool version is **1.7.0**. The tool is run with the following command-line options:

```
1 % ngram -lm model-file -order 5 -ppl queries-file \
2       -no-sos -no-eos -memuse -debug 0
```

No changes were done to the tool's source code.

KenLM

KenLM is a tool for estimating, filtering, and querying language models. The tool does not have clear version indication, so we used the tool's GitHub snapshot of the Git revision:

0f 306088c3d8b3a668c934f 605e21b693b959d4d

KenLM does not allow to switch off the probability reports from the command line. Therefore we had to modify the tool's code. In the `kenlm/lm/ngram query.hh` file we commented out the output code lines as follows:

```
1 struct BasicPrint {
2   void Word(StringPiece, WordIndex, const FullScoreReturn &) const {}
3   void Line(uint64_t oov, float total) const {
4     //std::cout << "Total: " << total << " OOV: " << oov << '\n';
5   }
6   void Summary(double, double, uint64_t, uint64_t) {}
7 };
8 struct FullPrint : public BasicPrint {
9   void Word(StringPiece surface, WordIndex vocab,
10            const FullScoreReturn &ret) const {
11     //std::cout << surface << '=' << vocab << ' '
12     //<< static_cast<unsigned int>(ret.ngram_length)
13     //<< ' ' << ret.prob << '\t';
14 }
15   void Summary(double ppl_including_oov, double ppl_excluding_oov,
16                uint64_t corpus_oov, uint64_t corpus_tokens) {
17     std::cout <<
18       "Perplexity including OOVs:\t" << ppl_including_oov << "\n"
19       "Perplexity excluding OOVs:\t" << ppl_excluding_oov << "\n"
20       "OOVs:\t" << corpus_oov << "\n"
21       "Tokens:\t" << corpus_tokens << '\n'
22       ;
23 } };
```

After this change, the tool was run with the following command-line options: 18

```
1 % query -n model-file < queries-file
```

Hardware configuration

The experiments were run on the following machine configuration:

```
1 [~ smt7 ~]$ lscpu
2 Architecture:          x86_64
3 CPU op-mode(s):        32-bit, 64-bit
4 Byte Order:            Little Endian
5 CPU(s):                40
6 On-line CPU(s) list:   0-39
7 Thread(s) per core:    2
8 Core(s) per socket:    10
9 Socket(s):             2
10 NUMA node(s):          2
11 Vendor ID:             GenuineIntel
12 CPU family:            6
13 Model:                 62
14 Stepping:              4
15 CPU MHz:               1200.000
16 BogoMIPS:              4999.23
17 Virtualization:        VT-x
18 L1d cache:             32K
19 L1i cache:             32K
20 L2 cache:              256K
21 L3 cache:              25600K
22 NUMA node0 CPU(s):     0-9,20-29
23 NUMA node1 CPU(s):     10-19,30-39
24 [~ smt7 ~]$ lsb_release -irc
25 Distributor ID: CentOS
26 Release:    6.7
27 Codename:   Final
28 [~ smt7 ~]$ grep MemTotal /proc/meminfo
29 MemTotal:       264496688 kB
```

Language models and query files

The considered language models and their sizes (in bytes) are:

```
1 [~ smt10~]$ ls -al *.lm
2 -rw-r--r-- 1     937792965 Sep 21 15:55 e_10_641093.lm
3 -rw-r--r-- 1    1708763123 Sep 21 17:36 e_20_1282186.lm
4 -rw-r--r-- 1    3148711562 Sep 21 17:45 e_30_2564372.lm
5 -rw-r--r-- 1    5880154140 Sep 21 18:09 e_40_5128745.lm
6 -rw-r--r-- 1   10952178505 Sep 21 18:29 e_50_10257490.lm
7 -rw-r--r-- 1   15667577793 Sep 21 20:22 e_60_15386235.lm
8 -rw-r--r-- 1   20098725535 Sep 21 20:37 e_70_20514981.lm
9 -rw-r--r-- 1   48998103628 Sep 21 21:08 e_80_48998103628.lm
```

The considered query files and their sizes are:

```
1 [~ smt10 ~]$ ls -al q_5_gram_1*.txt
2 -rw-r--r-- 1   2697064872 Sep 21 15:47 q_5_gram_100.000.000.txt
3 -rw-r--r-- 1           35 Sep 21 15:57 q_5_gram_1.txt
4 [~ smt10 ~]$
```

The number of m-grams per model is:

e_10_641093.lm

```
1 [~ smt10 ~]$ head -n 15 e_10_641093.lm
2 \data\
3 ngram 1=105682
4 ngram 2=1737132
5 ngram 3=5121040
6 ngram 4=7659442
7 ngram 5=8741158
```

e_20_1282186.lm

```
1 [~ smt10 ~]$ head -n 8 e_20_1282186.lm
2 \data\
3 ngram 1=143867
4 ngram 2=2707890
5 ngram 3=8886067
6 ngram 4=14188078
7 ngram 5=16757214
```

#####e_30_2564372.lm

```
1 [~ smt10 ~]$ head -n 8 e_30_2564372.lm
2 \data\
3 ngram 1=199164
4 ngram 2=4202658
5 ngram 3=15300577
6 ngram 4=26097321
7 ngram 5=31952150
```

e_40_5128745.lm

```
1 [~ smt10 ~]$ head -n 8 e_40_5128745.lm
2 \data\
3 ngram 1=298070
4 ngram 2=6675818
5 ngram 3=26819467
6 ngram 4=48897704
7 ngram 5=62194729
```

e_50_10257490.lm

```
1 [~ smt10 ~]$ head -n 8 e_50_10257490.lm
2 \data\
3 ngram 1=439499
4 ngram 2=10447874
5 ngram 3=46336705
6 ngram 4=90709359
7 ngram 5=120411272
```

e_60_15386235.lm

```
1 [~ smt10 ~]$ head -n 8 e_60_15386235.lm
2 \data\
3 ngram 1=568105
4 ngram 2=13574606
5 ngram 3=63474074
6 ngram 4=129430409
7 ngram 5=176283104
```

e_70_20514981.lm

```
1 [~ smt10 ~]$ head -n 8 e_70_20514981.lm
2 \data\
3 ngram 1=676750
4 ngram 2=16221298
5 ngram 3=78807519
6 ngram 4=165569280
7 ngram 5=229897626
```

e_80_48998103628.lm

```
1 [~ smt10 ~]$ head -n 8 e_80_48998103628.lm
2 \data\
3 ngram 1=2210728
4 ngram 2=67285057
5 ngram 3=183285165
6 ngram 4=396600722
7 ngram 5=563533665
```

Powered by Markdown-Cheatsheet

Chapter 2

Todo List

Member uva::smt::bpbd::server::decoder::de_configurator::allocate_decoder (acr_bool_flag is_stop, const string &source_sent, string &target_sent)

Pre-allocate decoders, make as many as there are threads

Parameters

is_stop	the flag that will be set to true in case one needs to abort the translation process.
source_sent	[in] the source language sentence to translate the source sentence is expected to be tokenized, reduced, and in the lower case.
target_sent	[out] the resulting target language sentence

Returns

an instance of the decoder object.

Member uva::smt::bpbd::server::decoder::de_configurator::dispose_decoder (sentence_decoder &dec)

Mark the decoder instance as available

Parameters

dec	the decoder to be returned

Class uva::smt::bpbd::server::lm::dictionary::counting_word_index

{Change or create a new version of the word index that will just use probabilities of the unigrams instead of counting words.}

Namespace uva::smt::bpbd::server::lm::identifiers

Go trough all the LM and change from TShordId and TLongId to phrase_uid, word_uid and basic types!

Member uva::smt::bpbd::server::lm::proxy::lm_proxy_local::allocate_fast_query_proxy ()

{In the future we should just use a number of stack allocated objects in order to reduce the new/delete overhead}

See also

lm_proxy

Member uva::smt::bpbd::server::lm::proxy::lm_proxy_local::allocate_slow_query_proxy ()

{In the future we should just use a number of stack allocated objects in order to reduce the new/delete overhead}

See also

lm_proxy

Member uva::smt::bpbd::server::lm::proxy::lm_proxy_local::dispose_fast_query_proxy (lm_fast_query_↩ proxy &query)

{In the future we should just use a number of stack allocated objects in order to reduce the new/delete overhead}

See also

lm_proxy

Member uva::smt::bpbd::server::lm::proxy::lm_proxy_local::dispose_slow_query_proxy (lm_slow_query↩ _proxy &query)

{In the future we should just use a number of stack allocated objects in order to reduce the new/delete overhead}

See also

lm_proxy

Member uva::smt::bpbd::server::rm::proxy::rm_proxy_local::allocate_query_proxy ()

{In the future we should just use a number of stack allocated objects in order to reduce the new/delete overhead}

See also

rm_proxy

Member uva::smt::bpbd::server::rm::proxy::rm_proxy_local::dispose_query_proxy (rm_query_proxy &query)

{In the future we should just use a number of stack allocated objects in order to reduce the new/delete overhead}

Parameters

query	the query to dispose

Member uva::smt::bpbd::server::rm::proxy::rm_proxy_local::load_model_data (char const ∗model_name, const rm_parameters &params)

Add the possibility to choose between the file readers from the command line!

Parameters

the	name of the model being loaded params the model parameters

Member uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features >::set_features (const size_t num_features, const prob_weight ∗features)

Get rid of magic constants here!

Parameters

num_features	the number of features to be set, already in the log10 scale
features	the weights to be set into the entry This is an array of translation weights, as we have here↩ : features[0] = p(f|e); features[1] = lex(p(f|e)); features[2] = p(e|f); features[3] = lex(p(e|f)); features[4] = phrase penalty; // optional

Member uva::smt::bpbd::server::tm::proxy::tm_proxy_local::dispose_query_proxy (tm_query_proxy &query)

In the future we should just use a number of stack allocated objects in order to reduce the new/delete overhead

See also

tm_proxy

Member uva::smt::bpbd::server::tm::proxy::tm_proxy_local::load_model_data (char const ∗model_name, const tm_parameters &params)

Add the possibility to choose between the file readers from the command line!

Parameters

the	name of the model being loaded params the model parameters

Member uva::smt::bpbd::server::trans_job::notify_task_done (const trans_task_ptr &task)

{Do a strict check on the tasks reporting to be finished, these should be the ones from the m_tasks list and they must report themselves only ones. (Optional - for safety).}

Parameters

task	the translation task that is finished

Member uva::smt::bpbd::server::trans_job_pool::add_job (trans_job_ptr trans_job)

{Later, the tasks pool shall be chosen based on the source and target language. This is for when a server can translate from multiple languages to multiple languages.}

Parameters

trans_job	the job to be added to the administration

Member uva::smt::bpbd::server::trans_manager::trans_manager (const size_t num_threads)

{Possibly limit the number of allowed open sessions (from one host and the maximum amount of allowed hosts) This is for later, if the server is put for www access.}

Member uva::smt::bpbd::server::trans_task_pool::notify_task_cancel (trans_task_ptr trans_task)

{To improve performance we could try checking if the tasks is already running, and if not then search the queue. Or use other data structure for a more efficient task removal. This is for the future, in case the performance is affected.}

Member uva::utils::containers::get_mem_incr_strat (const mem_inc_types_enum stype, const size_t min↩ _mem_inc, const size_t mem_inc_factor)

Optimize the switch, it is pretty ugly, use a map or something.

Parameters

stype	the strategy type
min_mem_inc	the minimum memory increment in number of elements
mem_inc_factor	the memory increment factor, the number we will multiply by the computed increment

Returns

the pointer to a newly allocated strategy object

Member uva::utils::logging::logger::set_reporting_level (const string level)

{This function is ugly improve it by using a map, or a similar so that we could just get an appropriate level for the string.}

Chapter 3

Namespace Index

3.1 Namespace List

Here is a list of all namespaces with brief descriptions:

uva . . . 37
uva::smt . . . 37
uva::smt::bpbd . . . 37
uva::smt::bpbd::client . . . 37
uva::smt::bpbd::common . . . 38
uva::smt::bpbd::common::messaging . . . 39
uva::smt::bpbd::common::messaging::job_id . . . 41
uva::smt::bpbd::common::messaging::session_id . . . 41
uva::smt::bpbd::server . . . 41
uva::smt::bpbd::server::common . . . 46
uva::smt::bpbd::server::common::models . . . 46
uva::smt::bpbd::server::decoder . . . 46
uva::smt::bpbd::server::decoder::sentence . . . 47
uva::smt::bpbd::server::decoder::stack . . . 47
uva::smt::bpbd::server::lm . . . 48
uva::smt::bpbd::server::lm::__C2DHybridTrie . . . 57
uva::smt::bpbd::server::lm::__C2DMapTrie . . . 57
uva::smt::bpbd::server::lm::__C2WArrayTrie . . . 57
uva::smt::bpbd::server::lm::__executor . . . 59
uva::smt::bpbd::server::lm::__G2DMapTrie . . . 59
uva::smt::bpbd::server::lm::__H2DMapTrie . . . 60
uva::smt::bpbd::server::lm::__LayeredTrieBase . . . 60
uva::smt::bpbd::server::lm::__W2CArrayTrie . . . 61
uva::smt::bpbd::server::lm::__W2CHybridTrie . . . 62
uva::smt::bpbd::server::lm::arpa . . . 62
uva::smt::bpbd::server::lm::caching . . . 63
uva::smt::bpbd::server::lm::dictionary . . . 63
uva::smt::bpbd::server::lm::dictionary::__AWordIndex . . . 64
uva::smt::bpbd::server::lm::dictionary::__counting_word_index . . . 64
uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index . . . 64
uva::smt::bpbd::server::lm::identifiers . . . 65
uva::smt::bpbd::server::lm::m_grams . . . 65
uva::smt::bpbd::server::lm::m_grams::m_gram_id . . . 66
uva::smt::bpbd::server::lm::proxy . . . 67
uva::smt::bpbd::server::rm . . . 67
uva::smt::bpbd::server::rm::builders . . . 68
uva::smt::bpbd::server::rm::models . . . 68
uva::smt::bpbd::server::rm::models::__rm_basic_model . . . 69

uva::smt::bpbd::server::rm::proxy . . . 69
uva::smt::bpbd::server::task_id . . . 69
uva::smt::bpbd::server::tm . . . 69
uva::smt::bpbd::server::tm::builders . . . 70
uva::smt::bpbd::server::tm::models . . . 70
uva::smt::bpbd::server::tm::models::__tm_basic_model . . . 71
uva::smt::bpbd::server::tm::proxy . . . 71
uva::utils . . . 71
uva::utils::containers . . . 72
uva::utils::containers::alloc . . . 74
uva::utils::containers::utils . . . 75
uva::utils::exceptions . . . 80
uva::utils::file . . . 80
uva::utils::hashing . . . 81
uva::utils::logging . . . 81
uva::utils::math . . . 83
uva::utils::math::bits . . . 83
uva::utils::math::const_expr . . . 83
uva::utils::math::log2 . . . 83
uva::utils::monitor . . . 83
uva::utils::text . . . 84
uva::utils::threads . . . 85

Chapter 4

Hierarchical Index

4.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

uva::smt::bpbd::server::lm::dictionary::aword_index . . . 89
uva::smt::bpbd::server::lm::dictionary::basic_word_index . . . 92
uva::smt::bpbd::server::lm::dictionary::counting_word_index . . . 131
uva::smt::bpbd::server::lm::dictionary::hashing_word_index . . . 173
uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type > . . . 227
uva::smt::bpbd::server::lm::caching::BitmapHashCache . . . 98
uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType > . . . 99
uva::utils::containers::circular_queue< elem_type, capacity > . . . 126
uva::smt::bpbd::client::client_config . . . 130
uva::smt::bpbd::server::decoder::de_configurator . . . 136
uva::smt::bpbd::server::decoder::de_parameters_struct . . . 138
uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR > . . . 140
uva::utils::containers::dynamic_stack_array< ARRAY_ELEM_TYPE, uint32_t > . . . 140
uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::WordDataEntry< ARRAY_ELEM_TYPE > . . . 384
uva::utils::containers::ELEMENT_DEALLOC_FUNC< ELEM_TYPE > . . . 144
exception
uva::utils::exceptions::uva_exception . . . 368
uva::utils::containers::fixed_size_hashmap< ELEMENT_TYPE, KEY_TYPE, IDX_TYPE > . . . 147
uva::utils::containers::alloc::greedy_memory_allocator< T > . . . 161
uva::utils::containers::greedy_memory_storage . . . 167
uva::smt::bpbd::common::messaging::id_manager< id_type > . . . 175
uva::smt::bpbd::common::messaging::id_manager< job_id_type > . . . 175
uva::smt::bpbd::common::messaging::id_manager< session_id_type > . . . 175
uva::smt::bpbd::common::messaging::id_manager< task_id_type > . . . 175
uva::smt::bpbd::server::lm::arpa::lm_basic_builder< trie_type, reader_type > . . . 180
uva::smt::bpbd::server::lm::lm_configurator . . . 182
uva::smt::bpbd::server::lm::__executor::lm_exec_params . . . 183
uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy . . . 184
uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type > . . . 186
uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight > . . . 190
uva::smt::bpbd::server::lm::arpa::lm_gram_builder_factory< TrieType > . . . 194
uva::smt::bpbd::server::lm::lm_parameters . . . 195
uva::smt::bpbd::server::lm::proxy::lm_proxy . . . 196
uva::smt::bpbd::server::lm::proxy::lm_proxy_local . . . 198
uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy . . . 201

uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local< trie_type > . . . 202
uva::utils::logging::logger . . . 206
uva::utils::logging::logging_synch . . . 209
uva::smt::bpbd::server::lm::m_grams::m_gram_payload_s . . . 209
uva::smt::bpbd::server::lm::m_gram_query . . . 210
uva::utils::containers::mem_increase_strategy . . . 217
uva::utils::monitor::memory_usage . . . 222
uva::smt::bpbd::server::decoder::stack::multi_stack . . . 226
uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_LENGTH > . . . 231
uva::smt::bpbd::server::lm::m_grams::phrase_base< MODEL_M_GRAM_MAX_LEN, MODEL_M_GRAM_MAX_LEN > . . . 231
uva::smt::bpbd::server::lm::m_grams::model_m_gram . . . 223
uva::smt::bpbd::server::lm::m_grams::phrase_base< QUERY_M_GRAM_MAX_LEN, LM_M_GRAM_LEVEL_MAX > . . . 231
uva::smt::bpbd::server::lm::m_grams::query_m_gram . . . 237
uva::smt::bpbd::server::decoder::sentence::phrase_data_entry . . . 236
uva::utils::containers::alloc::greedy_memory_allocator< T >::rebind< U > . . . 239
uva::smt::bpbd::server::rm::builders::rm_basic_builder< model_type, reader_type > . . . 239
uva::smt::bpbd::server::rm::models::rm_basic_model . . . 241
uva::smt::bpbd::server::rm::rm_configurator . . . 247
uva::smt::bpbd::server::rm::models::rm_entry_temp< num_features > . . . 248
uva::smt::bpbd::server::rm::rm_parameters . . . 251
uva::smt::bpbd::server::rm::proxy::rm_proxy . . . 252
uva::smt::bpbd::server::rm::proxy::rm_proxy_local . . . 253
uva::smt::bpbd::server::rm::models::rm_query< model_type > . . . 255
uva::smt::bpbd::server::rm::proxy::rm_query_proxy . . . 257
uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type > . . . 259
uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_GramData< TPayloadType > . . . 261
uva::smt::bpbd::server::lm::__W2CArrayTrie::S_M_GramData< PAYLOAD_TYPE > . . . 262
uva::smt::bpbd::server::lm::__G2DMapTrie::S_M_GramData< TPayloadType, TWordIdType > . . . 263
uva::smt::bpbd::server::decoder::sentence::sentence_decoder . . . 265
uva::smt::bpbd::server::server_parameters . . . 267
uva::smt::bpbd::server::decoder::stack::stack_data . . . 269
uva::smt::bpbd::server::decoder::stack::stack_level . . . 270
uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH > . . . 275
uva::utils::monitor::stat_monitor . . . 283
uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH > . . . 283
uva::smt::bpbd::server::lm::m_grams::m_gram_id::T_Gram_Id_Key . . . 288
uva::utils::containers::utils::T_IS_COMPARE_FUNC< ELEM_TYPE > . . . 288
uva::smt::bpbd::server::lm::arpa::TAddGramFunct< WordIndexType > . . . 289
uva::smt::bpbd::server::lm::__C2WArrayTrie::TCtxIdProbData . . . 289
uva::utils::file::text_piece_reader . . . 290
uva::utils::file::afile_reader . . . 87
uva::utils::file::cstyle_file_reader . . . 134
uva::utils::file::file_stream_reader . . . 145
uva::utils::file::memory_mapped_file_reader . . . 219
uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type > . . . 297
uva::smt::bpbd::server::tm::models::tm_basic_model . . . 301
uva::smt::bpbd::server::tm::tm_configurator . . . 306
uva::smt::bpbd::server::tm::tm_parameters . . . 307
uva::smt::bpbd::server::tm::proxy::tm_proxy . . . 308
uva::smt::bpbd::server::tm::proxy::tm_proxy_local . . . 310
uva::smt::bpbd::server::tm::models::tm_query< model_type > . . . 312

uva::smt::bpbd::server::tm::proxy::tm_query_proxy . . . 314
uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local< model_type > . . . 315
uva::smt::bpbd::server::tm::models::tm_source_entry . . . 317
uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features > . . . 320
uva::smt::bpbd::client::trans_job . . . 324
uva::smt::bpbd::server::trans_job . . . 325
uva::smt::bpbd::common::messaging::trans_job_code . . . 329
uva::smt::bpbd::server::trans_job_pool . . . 331
uva::smt::bpbd::common::messaging::trans_job_request . . . 336
uva::smt::bpbd::common::messaging::trans_job_response . . . 339
uva::smt::bpbd::client::trans_job_status . . . 342
uva::smt::bpbd::server::trans_manager . . . 344
uva::smt::bpbd::client::trans_manager . . . 347
uva::smt::bpbd::server::trans_task . . . 351
uva::smt::bpbd::server::trans_task_pool . . . 354
uva::smt::bpbd::server::trans_task_pool_worker . . . 357
uva::smt::bpbd::client::translation_client . . . 358
uva::smt::bpbd::server::translation_server . . . 361
uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::TSubArrReference . . . 364
uva::smt::bpbd::server::lm::__C2WArrayTrie::TWordIdPBData . . . 365
uva::smt::bpbd::server::lm::dictionary::__counting_word_index::TWordInfo . . . 365
uva::utils::containers::upp_diag_matrix< element_type > . . . 366
uva::utils::containers::upp_diag_matrix< phrase_data_entry > . . . 366
uva::smt::bpbd::server::lm::W2CH_UM_Storage . . . 378
uva::smt::bpbd::server::lm::W2CH_UM_StorageFactory< N > . . . 379
uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index::word_index_bucket_entry< word_id_type > . . . 381
uva::smt::bpbd::server::lm::word_index_trie_base< WordIndex > . . . 382
uva::smt::bpbd::server::lm::word_index_trie_base< lm_word_index > . . . 382
uva::smt::bpbd::server::lm::generic_trie_base< h2d_map_trie< lm_word_index >, lm_word_index, __H2DMapTrie::BITMAP_HASH_CACHE_BUCKETS_FACTOR > . . . 154
uva::smt::bpbd::server::lm::h2d_map_trie< lm_word_index > . . . 169
uva::smt::bpbd::server::lm::word_index_trie_base< WordIndexType > . . . 382
uva::smt::bpbd::server::lm::generic_trie_base< c2d_hybrid_trie< WordIndexType >, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR > . . . 154
uva::smt::bpbd::server::lm::layered_trie_base< c2d_hybrid_trie< WordIndexType >, WordIndexType, __C2DHybridTrie::BITMAP_HASH_CACHE_BUCKETS_FACTOR > . . . 176
uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType > . . . 115
uva::smt::bpbd::server::lm::generic_trie_base< c2d_map_trie< WordIndexType >, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR > . . . 154
uva::smt::bpbd::server::lm::layered_trie_base< c2d_map_trie< WordIndexType >, WordIndexType, __C2DMapTrie::BITMAP_HASH_CACHE_BUCKETS_FACTOR > . . . 176
uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType > . . . 118
uva::smt::bpbd::server::lm::generic_trie_base< c2w_array_trie< WordIndexType >, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR > . . . 154
uva::smt::bpbd::server::lm::layered_trie_base< c2w_array_trie< WordIndexType >, WordIndexType, __C2WArrayTrie::BITMAP_HASH_CACHE_BUCKETS_FACTOR > . . . 176
uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType > . . . 122
uva::smt::bpbd::server::lm::generic_trie_base< g2d_map_trie< WordIndexType >, WordIndexType, __G2DMapTrie::BITMAP_HASH_CACHE_BUCKETS_FACTOR > . . . 154
uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType > . . . 151
uva::smt::bpbd::server::lm::generic_trie_base< h2d_map_trie< WordIndexType >, WordIndexType, __H2DMapTrie::BITMAP_HASH_CACHE_BUCKETS_FACTOR > . . . 154
uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType > . . . 169
uva::smt::bpbd::server::lm::generic_trie_base< w2c_array_trie< WordIndexType >, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR > . . . 154

uva::smt::bpbd::server::lm::layered_trie_base< w2c_array_trie< WordIndexType >, WordIndex↩
Type, __W2CArrayTrie::BITMAP_HASH_CACHE_BUCKETS_FACTOR > 176
uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType > 370
uva::smt::bpbd::server::lm::generic_trie_base< w2c_hybrid_trie< WordIndexType, StorageFactory,
StorageContainer >, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR > . . 154
uva::smt::bpbd::server::lm::layered_trie_base< w2c_hybrid_trie< WordIndexType, Storage↩
Factory, StorageContainer >, WordIndexType, __W2CHybridTrie::BITMAP_HASH_CA↩
CHE_BUCKETS_FACTOR > 176
uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, Storage↩
Container > 374
uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE↩
_BUCKETS_FACTOR > 154
uva::smt::bpbd::server::lm::layered_trie_base< TrieType, WordIndexType, BITMAP_HASH_CA↩
CHE_BUCKETS_FACTOR > 176

Chapter 5

Class Index

5.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

uva::utils::file::afile_reader . . . 87
uva::smt::bpbd::server::lm::dictionary::aword_index . . . 89
uva::smt::bpbd::server::lm::dictionary::basic_word_index . . . 92
uva::smt::bpbd::server::lm::caching::BitmapHashCache . . . 98
uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType > . . . 99
uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType > . . . 115
uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType > . . . 118
uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType > . . . 122
uva::utils::containers::circular_queue< elem_type, capacity > . . . 126
uva::smt::bpbd::client::client_config . . . 130
uva::smt::bpbd::server::lm::dictionary::counting_word_index . . . 131
uva::utils::file::cstyle_file_reader . . . 134
uva::smt::bpbd::server::decoder::de_configurator . . . 136
uva::smt::bpbd::server::decoder::de_parameters_struct . . . 138
uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR > . . . 140
uva::utils::containers::ELEMENT_DEALLOC_FUNC< ELEM_TYPE > . . . 144
uva::utils::file::file_stream_reader . . . 145
uva::utils::containers::fixed_size_hashmap< ELEMENT_TYPE, KEY_TYPE, IDX_TYPE > . . . 147
uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType > . . . 151
uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR > . . . 154
uva::utils::containers::alloc::greedy_memory_allocator< T > . . . 161
uva::utils::containers::greedy_memory_storage . . . 167
uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType > . . . 169
uva::smt::bpbd::server::lm::dictionary::hashing_word_index . . . 173
uva::smt::bpbd::common::messaging::id_manager< id_type > . . . 175
uva::smt::bpbd::server::lm::layered_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR > . . . 176
uva::smt::bpbd::server::lm::arpa::lm_basic_builder< trie_type, reader_type > . . . 180
uva::smt::bpbd::server::lm::lm_configurator . . . 182
uva::smt::bpbd::server::lm::__executor::lm_exec_params . . . 183
uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy . . . 184
uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type > . . . 186
uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight > . . . 190
uva::smt::bpbd::server::lm::arpa::lm_gram_builder_factory< TrieType > . . . 194
uva::smt::bpbd::server::lm::lm_parameters . . . 195
uva::smt::bpbd::server::lm::proxy::lm_proxy . . . 196

uva::smt::bpbd::server::lm::proxy::lm_proxy_local . . . 198
uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy . . . 201
uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local< trie_type > . . . 202
uva::utils::logging::logger . . . 206
uva::utils::logging::logging_synch . . . 209
uva::smt::bpbd::server::lm::m_grams::m_gram_payload_s . . . 209
uva::smt::bpbd::server::lm::m_gram_query . . . 210
uva::utils::containers::mem_increase_strategy . . . 217
uva::utils::file::memory_mapped_file_reader . . . 219
uva::utils::monitor::memory_usage . . . 222
uva::smt::bpbd::server::lm::m_grams::model_m_gram . . . 223
uva::smt::bpbd::server::decoder::stack::multi_stack . . . 226
uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type > . . . 227
uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_L↩
ENGTH > . . . 231
uva::smt::bpbd::server::decoder::sentence::phrase_data_entry . . . 236
uva::smt::bpbd::server::lm::m_grams::query_m_gram . . . 237
uva::utils::containers::alloc::greedy_memory_allocator< T >::rebind< U > . . . 239
uva::smt::bpbd::server::rm::builders::rm_basic_builder< model_type, reader_type > . . . 239
uva::smt::bpbd::server::rm::models::rm_basic_model . . . 241
uva::smt::bpbd::server::rm::rm_configurator . . . 247
uva::smt::bpbd::server::rm::models::rm_entry_temp< num_features > . . . 248
uva::smt::bpbd::server::rm::rm_parameters . . . 251
uva::smt::bpbd::server::rm::proxy::rm_proxy . . . 252
uva::smt::bpbd::server::rm::proxy::rm_proxy_local . . . 253
uva::smt::bpbd::server::rm::models::rm_query< model_type > . . . 255
uva::smt::bpbd::server::rm::proxy::rm_query_proxy . . . 257
uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type > . . . 259
uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_GramData< TPayloadType > . . . 261
uva::smt::bpbd::server::lm::__W2CArrayTrie::S_M_GramData< PAYLOAD_TYPE > . . . 262
uva::smt::bpbd::server::lm::__G2DMapTrie::S_M_GramData< TPayloadType, TWordIdType > . . . 263
uva::smt::bpbd::server::decoder::sentence::sentence_decoder . . . 265
uva::smt::bpbd::server::server_parameters . . . 267
uva::smt::bpbd::server::decoder::stack::stack_data . . . 269
uva::smt::bpbd::server::decoder::stack::stack_level . . . 270
uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_↩
HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH > . . . 275
uva::utils::monitor::stat_monitor . . . 283
uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_H↩
ISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH > . . . 283
uva::smt::bpbd::server::lm::m_grams::m_gram_id::T_Gram_Id_Key . . . 288
uva::utils::containers::utils::T_IS_COMPARE_FUNC< ELEM_TYPE > . . . 288
uva::smt::bpbd::server::lm::arpa::TAddGramFunct< WordIndexType > . . . 289
uva::smt::bpbd::server::lm::__C2WArrayTrie::TCtxIdProbData . . . 289
uva::utils::file::text_piece_reader . . . 290
uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type > . . . 297
uva::smt::bpbd::server::tm::models::tm_basic_model . . . 301
uva::smt::bpbd::server::tm::tm_configurator . . . 306
uva::smt::bpbd::server::tm::tm_parameters . . . 307
uva::smt::bpbd::server::tm::proxy::tm_proxy . . . 308
uva::smt::bpbd::server::tm::proxy::tm_proxy_local . . . 310
uva::smt::bpbd::server::tm::models::tm_query< model_type > . . . 312
uva::smt::bpbd::server::tm::proxy::tm_query_proxy . . . 314
uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local< model_type > . . . 315
uva::smt::bpbd::server::tm::models::tm_source_entry . . . 317
uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features > . . . 320
uva::smt::bpbd::client::trans_job . . . 324
uva::smt::bpbd::server::trans_job . . . 325

uva::smt::bpbd::common::messaging::trans_job_code . . . 329
uva::smt::bpbd::server::trans_job_pool . . . 331
uva::smt::bpbd::common::messaging::trans_job_request . . . 336
uva::smt::bpbd::common::messaging::trans_job_response . . . 339
uva::smt::bpbd::client::trans_job_status . . . 342
uva::smt::bpbd::server::trans_manager . . . 344
uva::smt::bpbd::client::trans_manager . . . 347
uva::smt::bpbd::server::trans_task . . . 351
uva::smt::bpbd::server::trans_task_pool . . . 354
uva::smt::bpbd::server::trans_task_pool_worker . . . 357
uva::smt::bpbd::client::translation_client . . . 358
uva::smt::bpbd::server::translation_server . . . 361
uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::TSubArrReference . . . 364
uva::smt::bpbd::server::lm::__C2WArrayTrie::TWordIdPBData . . . 365
uva::smt::bpbd::server::lm::dictionary::__counting_word_index::TWordInfo . . . 365
uva::utils::containers::upp_diag_matrix< element_type > . . . 366
uva::utils::exceptions::uva_exception . . . 368
uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType > . . . 370
uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer > . . . 374
uva::smt::bpbd::server::lm::W2CH_UM_Storage . . . 378
uva::smt::bpbd::server::lm::W2CH_UM_StorageFactory< N > . . . 379
uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index::word_index_bucket_entry< word_id_type > . . . 381
uva::smt::bpbd::server::lm::word_index_trie_base< WordIndex > . . . 382
uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::WordDataEntry< ARRAY_ELEM_TYPE > . . . 384

Chapter 6

File Index

6.1 File List

Here is a list of all files with brief descriptions:

inc/main.hpp . . . 413
inc/client/client_config.hpp . . . 387
inc/client/trans_job.hpp . . . 387
inc/client/trans_job_status.hpp . . . 388
inc/client/trans_manager.hpp . . . 389
inc/client/translation_client.hpp . . . 390
inc/common/messaging/id_manager.hpp . . . 390
inc/common/messaging/trans_job_code.hpp . . . 391
inc/common/messaging/trans_job_id.hpp . . . 391
inc/common/messaging/trans_job_request.hpp . . . 392
inc/common/messaging/trans_job_response.hpp . . . 392
inc/common/messaging/trans_session_id.hpp . . . 393
inc/common/utils/exceptions.hpp . . . 400
inc/common/utils/hashing_utils.hpp . . . 404
inc/common/utils/math_utils.hpp . . . 410
inc/common/utils/string_utils.hpp . . . 412
inc/common/utils/threads.hpp . . . 413
inc/common/utils/containers/array_utils.hpp . . . 393
inc/common/utils/containers/circular_queue.hpp . . . 395
inc/common/utils/containers/dynamic_memory_arrays.hpp . . . 396
inc/common/utils/containers/fixed_size_hashmap.hpp . . . 398
inc/common/utils/containers/greedy_memory_allocator.hpp . . . 398
inc/common/utils/containers/greedy_memory_storage.hpp . . . 399
inc/common/utils/containers/upp_diag_matrix.hpp . . . 399
inc/common/utils/file/afile_reader.hpp . . . 401
inc/common/utils/file/cstyle_file_reader.hpp . . . 401
inc/common/utils/file/file_stream_reader.hpp . . . 402
inc/common/utils/file/memory_mapped_file_reader.hpp . . . 402
inc/common/utils/file/text_piece_reader.hpp . . . 403
inc/common/utils/logging/logger.hpp . . . 405
inc/common/utils/monitor/statistics_monitor.hpp . . . 411
inc/server/cmd_line_handler.hpp . . . 414
inc/server/server_configs.hpp . . . 450
inc/server/server_consts.hpp . . . 451
inc/server/server_parameters.hpp . . . 451
inc/server/trans_job.hpp . . . 388
inc/server/trans_job_pool.hpp . . . 458
inc/server/trans_manager.hpp . . . 389

inc/server/trans_task.hpp . . . 459
inc/server/trans_task_id.hpp . . . 459
inc/server/trans_task_pool.hpp . . . 459
inc/server/trans_task_pool_worker.hpp . . . 460
inc/server/translation_server.hpp . . . 460
inc/server/common/models/phrase_uid.hpp . . . 415
inc/server/decoder/de_configs.hpp . . . 415
inc/server/decoder/de_configurator.hpp . . . 416
inc/server/decoder/de_parameters.hpp . . . 416
inc/server/decoder/sentence/sentence_data_map.hpp . . . 416
inc/server/decoder/sentence/sentence_decoder.hpp . . . 417
inc/server/decoder/stack/multi_stack.hpp . . . 418
inc/server/decoder/stack/stack_data.hpp . . . 418
inc/server/decoder/stack/stack_level.hpp . . . 419
inc/server/decoder/stack/stack_state.hpp . . . 419
inc/server/decoder/stack/state_data.hpp . . . 420
inc/server/lm/lm_configs.hpp . . . 426
inc/server/lm/lm_configurator.hpp . . . 426
inc/server/lm/lm_consts.hpp . . . 427
inc/server/lm/lm_executor.hpp . . . 428
inc/server/lm/lm_parameters.hpp . . . 428
inc/server/lm/builders/lm_basic_builder.hpp . . . 420
inc/server/lm/builders/lm_gram_builder.hpp . . . 421
inc/server/lm/builders/lm_gram_builder_factory.hpp . . . 421
inc/server/lm/dictionaries/aword_index.hpp . . . 422
inc/server/lm/dictionaries/basic_word_index.hpp . . . 422
inc/server/lm/dictionaries/counting_word_index.hpp . . . 423
inc/server/lm/dictionaries/hashing_word_index.hpp . . . 423
inc/server/lm/dictionaries/optimizing_word_index.hpp . . . 424
inc/server/lm/mgrams/m_gram_id.hpp . . . 429
inc/server/lm/mgrams/m_gram_id_tables.hpp . . . 430
inc/server/lm/mgrams/m_gram_payload.hpp . . . 430
inc/server/lm/mgrams/model_m_gram.hpp . . . 431
inc/server/lm/mgrams/query_m_gram.hpp . . . 431
inc/server/lm/models/bitmap_hash_cache.hpp . . . 432
inc/server/lm/models/c2d_hybrid_trie.hpp . . . 432
inc/server/lm/models/c2d_map_trie.hpp . . . 433
inc/server/lm/models/c2w_array_trie.hpp . . . 433
inc/server/lm/models/g2d_map_trie.hpp . . . 434
inc/server/lm/models/generic_trie_base.hpp . . . 435
inc/server/lm/models/h2d_map_trie.hpp . . . 437
inc/server/lm/models/layered_trie_base.hpp . . . 437
inc/server/lm/models/m_gram_query.hpp . . . 439
inc/server/lm/models/w2c_array_trie.hpp . . . 439
inc/server/lm/models/w2c_hybrid_trie.hpp . . . 440
inc/server/lm/models/w2ch_um_storage.hpp . . . 441
inc/server/lm/models/word_index_trie_base.hpp . . . 441
inc/server/lm/proxy/lm_fast_query_proxy.hpp . . . 442
inc/server/lm/proxy/lm_fast_query_proxy_local.hpp . . . 442
inc/server/lm/proxy/lm_proxy.hpp . . . 443
inc/server/lm/proxy/lm_proxy_local.hpp . . . 443
inc/server/lm/proxy/lm_slow_query_proxy.hpp . . . 444
inc/server/lm/proxy/lm_slow_query_proxy_local.hpp . . . 444
inc/server/rm/rm_configs.hpp . . . 448
inc/server/rm/rm_configurator.hpp . . . 449
inc/server/rm/rm_consts.hpp . . . 449
inc/server/rm/rm_parameters.hpp . . . 450
inc/server/rm/builders/rm_basic_builder.hpp . . . 445

inc/server/rm/models/rm_basic_model.hpp . . . 445
inc/server/rm/models/rm_entry.hpp . . . 446
inc/server/rm/models/rm_query.hpp . . . 446
inc/server/rm/proxy/rm_proxy.hpp . . . 447
inc/server/rm/proxy/rm_proxy_local.hpp . . . 447
inc/server/rm/proxy/rm_query_proxy.hpp . . . 448
inc/server/rm/proxy/rm_query_proxy_local.hpp . . . 448
inc/server/tm/tm_configs.hpp . . . 456
inc/server/tm/tm_configurator.hpp . . . 457
inc/server/tm/tm_consts.hpp . . . 457
inc/server/tm/tm_parameters.hpp . . . 457
inc/server/tm/builders/tm_basic_builder.hpp . . . 452
inc/server/tm/models/tm_basic_model.hpp . . . 452
inc/server/tm/models/tm_query.hpp . . . 453
inc/server/tm/models/tm_source_entry.hpp . . . 453
inc/server/tm/models/tm_target_entry.hpp . . . 454
inc/server/tm/proxy/tm_proxy.hpp . . . 454
inc/server/tm/proxy/tm_proxy_local.hpp . . . 455
inc/server/tm/proxy/tm_query_proxy.hpp . . . 455
inc/server/tm/proxy/tm_query_proxy_local.hpp . . . 456
src/client/bpbd_client.cpp . . . 461
src/client/trans_job_status.cpp . . . 462
src/common/messaging/trans_job_code.cpp . . . 463
src/common/utils/logging/logger.cpp . . . 464
src/common/utils/monitor/statistics_monitor.cpp . . . 465
src/server/bpbd_server.cpp . . . 465
src/server/trans_task_pool.cpp . . . 477
src/server/trans_task_pool_worker.cpp . . . 478
src/server/decoder/de_configurator.cpp . . . 466
src/server/lm/lm_configurator.cpp . . . 469
src/server/lm/lm_query.cpp . . . 469
src/server/lm/builders/lm_basic_builder.cpp . . . 466
src/server/lm/builders/lm_gram_builder.cpp . . . 468
src/server/lm/mgrams/byte_m_gram_id.cpp . . . 470
src/server/lm/mgrams/model_m_gram.cpp . . . 471
src/server/lm/mgrams/query_m_gram.cpp . . . 471
src/server/lm/models/c2d_hybrid_trie.cpp . . . 472
src/server/lm/models/c2d_map_trie.cpp . . . 472
src/server/lm/models/c2w_array_trie.cpp . . . 473
src/server/lm/models/g2d_map_trie.cpp . . . 473
src/server/lm/models/h2d_map_trie.cpp . . . 474
src/server/lm/models/m_gram_query.cpp . . . 475
src/server/lm/models/w2c_array_trie.cpp . . . 475
src/server/lm/models/w2c_hybrid_trie.cpp . . . 476
src/server/rm/rm_configurator.cpp . . . 476
src/server/tm/tm_configurator.cpp . . . 477
src/server/tm/models/tm_target_entry.cpp . . . 477

Chapter 7

Namespace Documentation

7.1 uva Namespace Reference

Namespaces

- smt
- utils

7.2 uva::smt Namespace Reference

Namespaces

- bpbd

7.3 uva::smt::bpbd Namespace Reference

Namespaces

- client
- common
- server

7.4 uva::smt::bpbd::client Namespace Reference

Classes

- struct client_config
- struct trans_job
- class trans_job_status
- class trans_manager
- class translation_client

Typedefs

- typedef trans_job * trans_job_ptr

Functions

- ostream & operator<< (ostream &os, const trans_job_status &status)

7.4.1 Typedef Documentation

7.4.1.1 typedef trans_job* uva::smt::bpbd::client::trans_job_ptr

Definition at line 43 of file trans_job.hpp.

7.4.2 Function Documentation

7.4.2.1 ostream & uva::smt::bpbd::client::operator<< (ostream & *os,* const trans_job_status & *status*)

The stream output operator for the given translation job status instance

Parameters

os	the output stream
status	the status to be output

Returns

the output stream

Definition at line 57 of file trans_job_status.cpp.

7.5 uva::smt::bpbd::common Namespace Reference

Namespaces

- messaging

Functions

- template<typename INT_TYPE >
  INT_TYPE get_integer (INI<> &ini, string section, string key)
- string get_string (INI<> &ini, string section, string key)
- float get_float (INI<> &ini, string section, string key)

7.5.1 Function Documentation

7.5.1.1 float uva::smt::bpbd::common::get_float (INI<> & *ini,* string *section,* string *key*)

Definition at line 128 of file main.hpp.

7.5.1.2 template<typename INT_TYPE > INT_TYPE uva::smt::bpbd::common::get_integer (INI<> & *ini,* string *section,* string *key*)

Definition at line 105 of file main.hpp.

7.5.1.3 string uva::smt::bpbd::common::get_string (INI<> & *ini,* string *section,* string *key*)

Definition at line 120 of file main.hpp.

7.6 uva::smt::bpbd::common::messaging Namespace Reference

Namespaces

- job_id
- session_id

Classes

- class id_manager
- class trans_job_code
- class trans_job_request
- class trans_job_response

Typedefs

- typedef uint64_t job_id_type
- typedef trans_job_request * trans_job_request_ptr
- typedef trans_job_response * trans_job_response_ptr
- typedef uint64_t session_id_type

Functions

- ostream & operator<< (ostream &os, const trans_job_code &code)

7.6.1 Typedef Documentation

7.6.1.1 typedef uint64_t uva::smt::bpbd::common::messaging::job_id_type

Definition at line 36 of file trans_job_id.hpp.

7.6.1.2 typedef uint64_t uva::smt::bpbd::common::messaging::session_id_type

Definition at line 35 of file trans_session_id.hpp.

7.6.1.3 typedef trans_job_request* uva::smt::bpbd::common::messaging::trans_job_request_ptr

Definition at line 47 of file trans_job_request.hpp.

7.6.1.4 typedef trans_job_response* uva::smt::bpbd::common::messaging::trans_job_response_ptr

Definition at line 53 of file trans_job_response.hpp.

7.6.2 Function Documentation

7.6.2.1 ostream & uva::smt::bpbd::common::messaging::operator<< (ostream & *os*, const trans_job_code & *code*)

The stream output operator for the given translation job code instance

Parameters

os	the output stream
code	the code to be output

Returns

the output stream

Definition at line 59 of file trans_job_code.cpp.

7.7 uva::smt::bpbd::common::messaging::job_id Namespace Reference

7.8 uva::smt::bpbd::common::messaging::session_id Namespace Reference

7.9 uva::smt::bpbd::server Namespace Reference

Namespaces

- common
- decoder
- lm
- rm
- task_id
- tm

Classes

- struct server_parameters
- class trans_job
- class trans_job_pool
- class trans_manager
- class trans_task
- class trans_task_pool
- class trans_task_pool_worker
- class translation_server

Typedefs

- typedef uint16_t phrase_length
- typedef float prob_weight
- typedef uint64_t phrase_uid
- typedef uint64_t word_uid
- typedef trans_job * trans_job_ptr
- typedef trans_task * trans_task_ptr
- typedef uint64_t task_id_type

Functions

- void stop (translation_server &server, thread &server_thread)
- void print_the_prompt ()
- void print_server_commands ()
- bool begins_with (const string &str, const string &prefix)
- string get_string_value (const string &str, const string &prefix)
- int32_t get_int_value (const string &str, const string &prefix)
- float get_float_value (const string &str, const string &prefix)
- void set_log_level (const string &cmd, const string &prefix)
- void set_num_threads (server_parameters &params, translation_server &server, const string &cmd, const string &prefix)
- void set_decoder_params (const string &cmd, de_parameters &de_params)
- bool process_input_cmd (server_parameters &params, translation_server &server, thread &server_thread, char command[CMD_BUFF_SIZE])
- void perform_command_loop (server_parameters &params, translation_server &server, thread &server_↩ thread)

7.9.1 Typedef Documentation

7.9.1.1 typedef uint16_t uva::smt::bpbd::server::phrase_length

Definition at line 39 of file server_consts.hpp.

7.9.1.2 typedef uint64_t uva::smt::bpbd::server::phrase_uid

Definition at line 45 of file server_consts.hpp.

7.9.1.3 typedef float uva::smt::bpbd::server::prob_weight

Definition at line 42 of file server_consts.hpp.

7.9.1.4 typedef uint64_t uva::smt::bpbd::server::task_id_type

Definition at line 39 of file trans_task_id.hpp.

7.9.1.5 typedef trans_job* uva::smt::bpbd::server::trans_job_ptr

Definition at line 51 of file trans_job.hpp.

7.9.1.6 typedef trans_task * uva::smt::bpbd::server::trans_task_ptr

Definition at line 55 of file trans_task.hpp.

7.9.1.7 typedef uint64_t uva::smt::bpbd::server::word_uid

Definition at line 48 of file server_consts.hpp.

7.9.2 Function Documentation

7.9.2.1 bool uva::smt::bpbd::server::begins_with (const string & *str,* const string & *prefix*) [inline]

Allows to test if a string begins with a substring

Parameters

str	the string to check
the	prefix

Returns

true if the string begins with the prefix

Definition at line 100 of file cmd_line_handler.hpp.

7.9.2.2 float uva::smt::bpbd::server::get_float_value (const string & *str,* const string & *prefix*) [inline]

Allows to parse the command parameter and return it

Parameters

str	the command string
prefix	the command pregix

Returns

the parsed value

Definition at line 139 of file cmd_line_handler.hpp.

7.9.2.3 int32_t uva::smt::bpbd::server::get_int_value (const string & *str,* const string & *prefix*) [inline]

Allows to parse the command parameter and return it

Parameters

str	the command string
prefix	the command pregix

Returns

the parsed value

Definition at line 120 of file cmd_line_handler.hpp.

7.9.2.4 string uva::smt::bpbd::server::get_string_value (const string & *str,* const string & *prefix*) [inline]

Allows to parse the command parameter and return it as a string

Parameters

str	the command string
prefix	the command pregix

Returns

the parsed value

Definition at line 110 of file cmd_line_handler.hpp.

7.9.2.5 void uva::smt::bpbd::server::perform_command_loop (server_parameters & *params,* translation_server & *server,* thread & *server_thread*)

Runs the server's command loop

Parameters

params	some server params
server	the server being run
server_thread	the server thread

Definition at line 311 of file cmd_line_handler.hpp.

7.9.2.6 void uva::smt::bpbd::server::print_server_commands ( )

Prints the available server commands

Definition at line 74 of file cmd_line_handler.hpp.

7.9.2.7 void uva::smt::bpbd::server::print_the_prompt ( )

Allows to print the prompt

Definition at line 67 of file cmd_line_handler.hpp.

7.9.2.8 bool uva::smt::bpbd::server::process_input_cmd (server_parameters & *params,* translation_server & *server,* thread & *server_thread,* char *command[CMD_BUFF_SIZE]*) [inline]

Allowsto process the command params some server parameters

Parameters

server	the server being run
server_thread	the server thread
command	the command sting to handle

Returns

true if we need to stop, otherwise false

Definition at line 251 of file cmd_line_handler.hpp.

7.9.2.9 void uva::smt::bpbd::server::set_decoder_params (const string & *cmd,* de_parameters & *de_params*) [inline]

Allows to set some decoder parameters

Parameters

cmd	the command to process, if not a command for setting decoder parameters an error will be reported.
m_de_params	the reference to the decoder parameters to set with new values.

Definition at line 196 of file cmd_line_handler.hpp.

7.9.2.10 void uva::smt::bpbd::server::set_log_level (const string & *cmd,* const string & *prefix*) [inline]

Allows to set the debug level

Parameters

cmd	the debug level

Definition at line 156 of file cmd_line_handler.hpp.

7.9.2.11 void uva::smt::bpbd::server::set_num_threads (server_parameters & *params*, translation_server & *server*, const string & *cmd*, const string & *prefix*) [inline]

Allows to set the number of worker threads

Parameters

params	the server parameters
server	the translation server
cmd	the input command
prefix	the command prefix

Definition at line 167 of file cmd_line_handler.hpp.

7.9.2.12 void uva::smt::bpbd::server::stop (translation_server & *server*, thread & *server_thread*)

Allows to stop the server;

Parameters

server	the server being run
server_thread	the server thread

Definition at line 53 of file cmd_line_handler.hpp.

7.10 uva::smt::bpbd::server::common Namespace Reference

Namespaces

- models

7.11 uva::smt::bpbd::server::common::models Namespace Reference

7.12 uva::smt::bpbd::server::decoder Namespace Reference

Namespaces

- sentence
- stack

Classes

- class de_configurator
- struct de_parameters_struct

Typedefs

- typedef de_parameters_struct de_parameters

7.12.1 Typedef Documentation

7.12.1.1 typedef de_parameters_struct uva::smt::bpbd::server::decoder::de_parameters

Definition at line 174 of file de_parameters.hpp.

7.13 uva::smt::bpbd::server::decoder::sentence Namespace Reference

Classes

- struct phrase_data_entry
- class sentence_decoder

Typedefs

- typedef upp_diag_matrix< phrase_data_entry > sentence_data_map

7.13.1 Typedef Documentation

7.13.1.1 typedef upp_diag_matrix<phrase_data_entry> uva::smt::bpbd::server::decoder::sentence↩ ::sentence_data_map

Definition at line 99 of file sentence_data_map.hpp.

7.14 uva::smt::bpbd::server::decoder::stack Namespace Reference

Classes

- class multi_stack
- struct stack_data
- class stack_level
- class stack_state_templ
- struct state_data_templ

Typedefs

- typedef stack_state_templ< MAX_WORDS_PER_SENTENCE, LM_HISTORY_LEN_MAX, LM_MAX_QU↩ ERY_LEN > stack_state
- typedef stack_state * stack_state_ptr
- typedef function< void(stack_state_ptr) > add_new_state_function
- typedef stack_level * stack_level_ptr

7.14.1 Typedef Documentation

7.14.1.1 typedef function<void(stack_state_ptr) > uva::smt::bpbd::server::decoder::stack::add_new_state_↩ function

Definition at line 47 of file stack_data.hpp.

7.14.1.2 typedef stack_level* uva::smt::bpbd::server::decoder::stack::stack_level_ptr

Definition at line 38 of file stack_level.hpp.

7.14.1.3 typedef stack_state_templ<MAX_WORDS_PER_SENTENCE, LM_HISTORY_LEN_MAX, LM_MAX_QUERY_LEN> uva::smt::bpbd::server::decoder::stack::stack_state

Definition at line 37 of file stack_data.hpp.

7.14.1.4 typedef stack_state* uva::smt::bpbd::server::decoder::stack::stack_state_ptr

Definition at line 44 of file stack_data.hpp.

7.15 uva::smt::bpbd::server::lm Namespace Reference

Namespaces

- __C2DHybridTrie
- __C2DMapTrie
- __C2WArrayTrie
- __executor
- __G2DMapTrie
- __H2DMapTrie
- __LayeredTrieBase
- __W2CArrayTrie
- __W2CHybridTrie
- arpa
- caching
- dictionary
- identifiers
- m_grams
- proxy

Classes

- class c2d_hybrid_trie
- class c2d_map_trie
- class c2w_array_trie
- class g2d_map_trie
- class generic_trie_base
- class h2d_map_trie
- class layered_trie_base
- class lm_configurator
- struct lm_parameters
- class m_gram_query
- class w2c_array_trie
- class w2c_hybrid_trie
- class W2CH_UM_Storage
- class W2CH_UM_StorageFactory
- class word_index_trie_base

Typedefs

- typedef hashing_word_index lm_word_index
- typedef h2d_map_trie< lm_word_index > lm_model_type
- typedef cstyle_file_reader lm_model_reader
- typedef lm_basic_builder< lm_model_type, lm_model_reader > lm_builder_type
- typedef c2d_hybrid_trie< basic_word_index > TC2DHybridTrieBasic
- typedef c2d_hybrid_trie< counting_word_index > TC2DHybridTrieCount
- typedef c2d_hybrid_trie< basic_optimizing_word_index > TC2DHybridTrieOptBasic
- typedef c2d_hybrid_trie< counting_optimizing_word_index > TC2DHybridTrieOptCount
- typedef c2d_hybrid_trie< hashing_word_index > TC2DHybridTrieHashing
- typedef c2d_map_trie< basic_word_index > TC2DMapTrieBasic
- typedef c2d_map_trie< counting_word_index > TC2DMapTrieCount
- typedef c2d_map_trie< hashing_word_index > TC2DMapTrieHashing
- typedef c2d_map_trie< basic_optimizing_word_index > TC2DMapTrieOptBasic
- typedef c2d_map_trie< counting_optimizing_word_index > TC2DMapTrieOptCount
- typedef c2w_array_trie< basic_word_index > TC2WArrayTrieBasic
- typedef c2w_array_trie< counting_word_index > TC2WArrayTrieCount
- typedef c2w_array_trie< basic_optimizing_word_index > TC2WArrayTrieOptBasic
- typedef c2w_array_trie< counting_optimizing_word_index > TC2WArrayTrieOptCount
- typedef c2w_array_trie< hashing_word_index > TC2WArrayTrieHashing
- typedef g2d_map_trie< basic_word_index > TG2DMapTrieBasic
- typedef g2d_map_trie< counting_word_index > TG2DMapTrieCount
- typedef g2d_map_trie< basic_optimizing_word_index > TG2DMapTrieOptBasic
- typedef g2d_map_trie< counting_optimizing_word_index > TG2DMapTrieOptCount
- typedef g2d_map_trie< hashing_word_index > TG2DMapTrieHashing
- typedef h2d_map_trie< basic_word_index > TH2DMapTrieBasic
- typedef h2d_map_trie< counting_word_index > TH2DMapTrieCount
- typedef h2d_map_trie< basic_optimizing_word_index > TH2DMapTrieOptBasic
- typedef h2d_map_trie< counting_optimizing_word_index > TH2DMapTrieOptCount
- typedef h2d_map_trie< hashing_word_index > TH2DMapTrieHashing
- typedef w2c_array_trie< basic_word_index > TW2CArrayTrieBasic
- typedef w2c_array_trie< counting_word_index > TW2CArrayTrieCount
- typedef w2c_array_trie< basic_optimizing_word_index > TW2CArrayTrieOptBasic
- typedef w2c_array_trie< counting_optimizing_word_index > TW2CArrayTrieOptCount
- typedef w2c_array_trie< hashing_word_index > TW2CArrayTrieHashing
- typedef w2c_hybrid_trie< basic_word_index > TW2CHybridTrieBasic
- typedef w2c_hybrid_trie< counting_word_index > TW2CHybridTrieCount
- typedef w2c_hybrid_trie< basic_optimizing_word_index > TW2CHybridTrieOptBasic
- typedef w2c_hybrid_trie< counting_optimizing_word_index > TW2CHybridTrieOptCount
- typedef w2c_hybrid_trie< hashing_word_index > TW2CHybridTrieHashing
- typedef pair< const TShortId, TShortId > TStorageMapEntry
- typedef greedy_memory_allocator< TStorageMapEntry > TStorageMapAllocator
- typedef unordered_map< TShortId, TShortId, std::hash< TShortId >, std::equal_to< TShortId >, TStorageMapAllocator > TStorageUnsignedMap
- typedef map< TShortId, TShortId > TStorageMap

Enumerations

- enum MGramStatusEnum { UNDEFINED_MGS = 0, BAD_END_WORD_UNKNOWN_MGS = 1, BAD_NO_PAYLOAD_MGS = 2, GOOD_PRESENT_MGS = 3 }

Functions

- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_hybrid_trie, basic_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_hybrid_trie, counting_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_hybrid_trie, hashing_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_hybrid_trie, basic_optimizing_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_hybrid_trie, counting_optimizing_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_map_trie, basic_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_map_trie, counting_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_map_trie, hashing_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_map_trie, basic_optimizing_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_map_trie, counting_optimizing_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2w_array_trie, basic_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2w_array_trie, counting_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2w_array_trie, hashing_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2w_array_trie, basic_optimizing_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2w_array_trie, counting_optimizing_word_index)
- INSTANTIATE_TRIE_TEMPLATE_TYPE (g2d_map_trie, basic_word_index)
- INSTANTIATE_TRIE_TEMPLATE_TYPE (g2d_map_trie, counting_word_index)
- INSTANTIATE_TRIE_TEMPLATE_TYPE (g2d_map_trie, hashing_word_index)
- INSTANTIATE_TRIE_TEMPLATE_TYPE (g2d_map_trie, basic_optimizing_word_index)
- INSTANTIATE_TRIE_TEMPLATE_TYPE (g2d_map_trie, counting_optimizing_word_index)
- INSTANTIATE_TRIE_TEMPLATE_TYPE (h2d_map_trie, basic_word_index)
- INSTANTIATE_TRIE_TEMPLATE_TYPE (h2d_map_trie, counting_word_index)
- INSTANTIATE_TRIE_TEMPLATE_TYPE (h2d_map_trie, hashing_word_index)
- INSTANTIATE_TRIE_TEMPLATE_TYPE (h2d_map_trie, basic_optimizing_word_index)
- INSTANTIATE_TRIE_TEMPLATE_TYPE (h2d_map_trie, counting_optimizing_word_index)
- ostream & operator<< (ostream &stream, const m_gram_query &query)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_array_trie, basic_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_array_trie, counting_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_array_trie, hashing_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_array_trie, basic_optimizing_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_array_trie, counting_optimizing_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_hybrid_trie, basic_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_hybrid_trie, counting_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_hybrid_trie, hashing_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_hybrid_trie, basic_optimizing_word_index)
- INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_hybrid_trie, counting_optimizing_word_index)

Variables

- class uva::smt::bpbd::server::lm::lm_configurator __attribute__
- const prob_weight DEF_UNK_WORD_LOG_PROB_WEIGHT = -10.0f

7.15.1 Typedef Documentation

7.15.1.1 typedef lm_basic_builder<lm_model_type, lm_model_reader> uva::smt::bpbd::server::lm::lm_builder_type

Definition at line 78 of file lm_configs.hpp.

7.15.1.2 typedef cstyle_file_reader uva::smt::bpbd::server::lm::lm_model_reader

Definition at line 75 of file lm_configs.hpp.

7.15.1.3 typedef h2d_map_trie<lm_word_index> uva::smt::bpbd::server::lm::lm_model_type

Definition at line 72 of file lm_configs.hpp.

7.15.1.4 typedef hashing_word_index uva::smt::bpbd::server::lm::lm_word_index

Definition at line 69 of file lm_configs.hpp.

7.15.1.5 typedef c2d_hybrid_trie< basic_word_index > uva::smt::bpbd::server::lm::TC2DHybridTrieBasic

Definition at line 346 of file c2d_hybrid_trie.hpp.

7.15.1.6 typedef c2d_hybrid_trie< counting_word_index > uva::smt::bpbd::server::lm::TC2DHybridTrieCount

Definition at line 347 of file c2d_hybrid_trie.hpp.

7.15.1.7 typedef c2d_hybrid_trie< hashing_word_index > uva::smt::bpbd::server::lm::TC2DHybridTrieHashing

Definition at line 350 of file c2d_hybrid_trie.hpp.

7.15.1.8 typedef c2d_hybrid_trie< basic_optimizing_word_index > uva::smt::bpbd::server::lm::TC2DHybridTrieOptBasic

Definition at line 348 of file c2d_hybrid_trie.hpp.

7.15.1.9 typedef c2d_hybrid_trie< counting_optimizing_word_index > uva::smt::bpbd::server::lm::TC2DHybridTrieOptCount

Definition at line 349 of file c2d_hybrid_trie.hpp.

7.15.1.10 typedef c2d_map_trie<basic_word_index > uva::smt::bpbd::server::lm::TC2DMapTrieBasic

Definition at line 355 of file c2d_map_trie.hpp.

7.15.1.11 typedef c2d_map_trie<counting_word_index > uva::smt::bpbd::server::lm::TC2DMapTrieCount

Definition at line 356 of file c2d_map_trie.hpp.

7.15.1.12 typedef c2d_map_trie<hashing_word_index > uva::smt::bpbd::server::lm::TC2DMapTrieHashing

Definition at line 357 of file c2d_map_trie.hpp.

7.15.1.13 typedef c2d_map_trie<basic_optimizing_word_index > uva::smt::bpbd::server::lm::TC2DMap↩ TrieOptBasic

Definition at line 358 of file c2d_map_trie.hpp.

7.15.1.14 typedef c2d_map_trie<counting_optimizing_word_index > uva::smt::bpbd::server::lm::TC2D↩ MapTrieOptCount

Definition at line 359 of file c2d_map_trie.hpp.

7.15.1.15 typedef c2w_array_trie<basic_word_index > uva::smt::bpbd::server::lm::TC2WArrayTrieBasic

Definition at line 484 of file c2w_array_trie.hpp.

7.15.1.16 typedef c2w_array_trie<counting_word_index > uva::smt::bpbd::server::lm::TC2WArrayTrieCount

Definition at line 485 of file c2w_array_trie.hpp.

7.15.1.17 typedef c2w_array_trie<hashing_word_index > uva::smt::bpbd::server::lm::TC2WArrayTrie↩ Hashing

Definition at line 488 of file c2w_array_trie.hpp.

7.15.1.18 typedef c2w_array_trie<basic_optimizing_word_index > uva::smt::bpbd::server::lm::TC2W↩ ArrayTrieOptBasic

Definition at line 486 of file c2w_array_trie.hpp.

7.15.1.19 typedef c2w_array_trie<counting_optimizing_word_index > uva::smt::bpbd::server::lm::TC2W↩ ArrayTrieOptCount

Definition at line 487 of file c2w_array_trie.hpp.

7.15.1.20 typedef g2d_map_trie<basic_word_index > uva::smt::bpbd::server::lm::TG2DMapTrieBasic

Definition at line 300 of file g2d_map_trie.hpp.

7.15.1.21 typedef g2d_map_trie<counting_word_index > uva::smt::bpbd::server::lm::TG2DMapTrieCount

Definition at line 301 of file g2d_map_trie.hpp.

7.15.1.22 typedef g2d_map_trie<hashing_word_index > uva::smt::bpbd::server::lm::TG2DMapTrieHashing

Definition at line 304 of file g2d_map_trie.hpp.

7.15.1.23 typedef g2d_map_trie<basic_optimizing_word_index > uva::smt::bpbd::server::lm::TG2DMap↩ TrieOptBasic

Definition at line 302 of file g2d_map_trie.hpp.

7.15.1.24 typedef g2d_map_trie<counting_optimizing_word_index > uva::smt::bpbd::server::lm::TG2D↩ MapTrieOptCount

Definition at line 303 of file g2d_map_trie.hpp.

7.15.1.25 typedef h2d_map_trie<basic_word_index > uva::smt::bpbd::server::lm::TH2DMapTrieBasic

Definition at line 300 of file h2d_map_trie.hpp.

7.15.1.26 typedef h2d_map_trie<counting_word_index > uva::smt::bpbd::server::lm::TH2DMapTrieCount

Definition at line 301 of file h2d_map_trie.hpp.

7.15.1.27 typedef h2d_map_trie<hashing_word_index > uva::smt::bpbd::server::lm::TH2DMapTrieHashing

Definition at line 304 of file h2d_map_trie.hpp.

7.15.1.28 typedef h2d_map_trie<basic_optimizing_word_index > uva::smt::bpbd::server::lm::TH2DMap↩ TrieOptBasic

Definition at line 302 of file h2d_map_trie.hpp.

7.15.1.29 typedef h2d_map_trie<counting_optimizing_word_index > uva::smt::bpbd::server::lm::TH2D↩ MapTrieOptCount

Definition at line 303 of file h2d_map_trie.hpp.

7.15.1.30 typedef map<TShortId, TShortId> uva::smt::bpbd::server::lm::TStorageMap

Definition at line 56 of file w2ch_um_storage.hpp.

7.15.1.31 typedef greedy_memory_allocator< TStorageMapEntry > uva::smt::bpbd::server::lm::TStorage↩ MapAllocator

Definition at line 52 of file w2ch_um_storage.hpp.

7.15.1.32 typedef pair< const TShortId, TShortId> uva::smt::bpbd::server::lm::TStorageMapEntry

Definition at line 50 of file w2ch_um_storage.hpp.

7.15.1.33 typedef unordered_map<TShortId, TShortId, std::hash<TShortId>, std::equal_to<TShortId>, TStorageMapAllocator > uva::smt::bpbd::server::lm::TStorageUnsignedMap

Definition at line 54 of file w2ch_um_storage.hpp.

7.15.1.34 typedef w2c_array_trie<basic_word_index > uva::smt::bpbd::server::lm::TW2CArrayTrieBasic

Definition at line 570 of file w2c_array_trie.hpp.

7.15.1.35 typedef w2c_array_trie<counting_word_index > uva::smt::bpbd::server::lm::TW2CArrayTrieCount

Definition at line 571 of file w2c_array_trie.hpp.

7.15.1.36 typedef w2c_array_trie<hashing_word_index > uva::smt::bpbd::server::lm::TW2CArrayTrie↩ Hashing

Definition at line 574 of file w2c_array_trie.hpp.

7.15.1.37 typedef w2c_array_trie<basic_optimizing_word_index > uva::smt::bpbd::server::lm::TW2C↩ ArrayTrieOptBasic

Definition at line 572 of file w2c_array_trie.hpp.

7.15.1.38 typedef w2c_array_trie<counting_optimizing_word_index > uva::smt::bpbd::server::lm::TW2C↩ ArrayTrieOptCount

Definition at line 573 of file w2c_array_trie.hpp.

7.15.1.39 typedef w2c_hybrid_trie<basic_word_index> uva::smt::bpbd::server::lm::TW2CHybridTrieBasic

Definition at line 323 of file w2c_hybrid_trie.hpp.

7.15.1.40 typedef w2c_hybrid_trie<counting_word_index> uva::smt::bpbd::server::lm::TW2CHybridTrie↩ Count

Definition at line 324 of file w2c_hybrid_trie.hpp.

7.15.1.41 typedef w2c_hybrid_trie<hashing_word_index> uva::smt::bpbd::server::lm::TW2CHybridTrie↩ Hashing

Definition at line 327 of file w2c_hybrid_trie.hpp.

7.15.1.42 typedef w2c_hybrid_trie<basic_optimizing_word_index> uva::smt::bpbd::server::lm::TW2C↩ HybridTrieOptBasic

Definition at line 325 of file w2c_hybrid_trie.hpp.

7.15.1.43 typedef w2c_hybrid_trie<counting_optimizing_word_index> uva::smt::bpbd::server::lm::TW2C↩ HybridTrieOptCount

Definition at line 326 of file w2c_hybrid_trie.hpp.

7.15.2 Enumeration Type Documentation

7.15.2.1 enum uva::smt::bpbd::server::lm::MGramStatusEnum

Contains the m-gram status values: 0. UNDEFINED_MGS - the status is undefined

1. BAD_END_WORD_UNKNOWN_MGS - the m-gram is definitely not present the end word is unknown
2. BAD_NO_PAYLOAD_MGS - the m-gram is definitely not present, the m-gram hash is not cached, or it is not found in the trie (the meaning depends on the context)
3. GOOD_PRESENT_MGS - the m-gram is potentially present, its hash is cached, or it is found in the trie (the meaning depends on the context)

Enumerator

UNDEFINED_MGS

BAD_END_WORD_UNKNOWN_MGS

BAD_NO_PAYLOAD_MGS

GOOD_PRESENT_MGS

Definition at line 81 of file generic_trie_base.hpp.

7.15.3 Function Documentation

7.15.3.1 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_array_trie , basic_word_index)

7.15.3.2 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_array_trie , counting_word_index)

7.15.3.3 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_array_trie , hashing_word_index)

7.15.3.4 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_array_trie , basic_optimizing_word_index)

7.15.3.5 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_array_trie , counting_optimizing_word_index)

7.15.3.6 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2w_array_trie , basic_word_index)

7.15.3.7 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2w_array_trie , counting_word_index)

7.15.3.8 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2w_array_trie , hashing_word_index)

7.15.3.9 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2w_array_trie , basic_optimizing_word_index)

7.15.3.10 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2w_array_trie , counting_optimizing_word_index)

7.15.3.11 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_hybrid_trie , basic_word_index)

7.15.3.12 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_hybrid_trie , counting_word_index)

7.15.3.13 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_hybrid_trie , hashing_word_index)

7.15.3.14 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_hybrid_trie , basic_optimizing_word_index)

7.15.3.15 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_hybrid_trie , counting_optimizing_word_index)

7.15.3.16 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_map_trie , basic_word_index)

7.15.3.17 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_map_trie , counting_word_index)

7.15.3.18 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_map_trie , hashing_word_index)

7.15.3.19 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_map_trie , basic_optimizing_word_index)

7.15.3.20 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_map_trie , counting_optimizing_word_index)

7.15.3.21 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_hybrid_trie , basic_word_index)

7.15.3.22 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_hybrid_trie , counting_word_index)

7.15.3.23 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_hybrid_trie , hashing_word_index)

7.15.3.24 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_hybrid_trie , basic_optimizing_word_index)

7.15.3.25 uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_hybrid_trie , counting_optimizing_word_index)

7.15.3.26 uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (h2d_map_trie , basic_word_index)

7.15.3.27 uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (h2d_map_trie , counting_word_index)

7.15.3.28 uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (h2d_map_trie , hashing_word_index)

7.15.3.29 uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (h2d_map_trie , basic_optimizing_word_index)

7.15.3.30 uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (h2d_map_trie , counting_optimizing_word_index)

7.15.3.31 uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (g2d_map_trie , basic_word_index)

7.15.3.32 uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (g2d_map_trie , counting_word_index)

7.15.3.33 uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (g2d_map_trie , hashing_word_index)

7.15.3.34 uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (g2d_map_trie , basic_optimizing_word_index)

7.15.3.35 uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (g2d_map_trie , counting_optimizing_word_index)

7.15.3.36 ostream& uva::smt::bpbd::server::lm::operator<< (ostream & *stream*, const m_gram_query & *query*)

Allows to serialize the m-gram query to the output stream as a string

Parameters

stream	the reference to the stream to output into
query	the query object to output

Returns

the reference to the stream

Definition at line 40 of file m_gram_query.cpp.

7.15.4 Variable Documentation

7.15.4.1 class uva::smt::bpbd::server::lm::lm_configurator uva::smt::bpbd::server::lm::__attribute__

7.15.4.2 const prob_weight uva::smt::bpbd::server::lm::DEF_UNK_WORD_LOG_PROB_WEIGHT = -10.0f

Definition at line 68 of file server_configs.hpp.

7.16 uva::smt::bpbd::server::lm::__C2DHybridTrie Namespace Reference

7.17 uva::smt::bpbd::server::lm::__C2DMapTrie Namespace Reference

7.18 uva::smt::bpbd::server::lm::__C2WArrayTrie Namespace Reference

Classes

- struct TCtxIdProbData
- struct TWordIdPBData

Functions

- bool operator< (const TWordIdPBData &one, const TWordIdPBData &two)
- int8_t compare (const TCtxIdProbData &one, const TCtxIdProbData &two)
- bool operator< (const TCtxIdProbData &one, const TCtxIdProbData &two)
- bool operator> (const TCtxIdProbData &one, const TCtxIdProbData &two)
- bool operator== (const TCtxIdProbData &one, const TCtxIdProbData &two)

7.18.1 Function Documentation

7.18.1.1 int8_t uva::smt::bpbd::server::lm::__C2WArrayTrie::compare (const TCtxIdProbData & *one,* const TCtxIdProbData & *two*) [inline]

This is the compare operator implementation

Parameters

one	the first object to compare
two	the second object to compare

Returns

-1 if (word_id,ctx_id) $<$ (word_id,ctx_id) 0 if (word_id,ctx_id) == (word_id,ctx_id) +1 if (word_id,ctx_id) $>$ (word_id,ctx_id)

Definition at line 95 of file c2w_array_trie.hpp.

7.18.1.2 bool uva::smt::bpbd::server::lm::__C2WArrayTrie::operator< (const TWordIdPBData & *one*, const TWordIdPBData & *two*) [inline]

This is the less operator implementation

Parameters

one	the first object to compare
two	the second object to compare

Returns

true one.id $<$ two.id

Definition at line 69 of file c2w_array_trie.hpp.

7.18.1.3 bool uva::smt::bpbd::server::lm::__C2WArrayTrie::operator< (const TCtxIdProbData & *one*, const TCtxIdProbData & *two*) [inline]

Definition at line 120 of file c2w_array_trie.hpp.

7.18.1.4 bool uva::smt::bpbd::server::lm::__C2WArrayTrie::operator== (const TCtxIdProbData & *one*, const TCtxIdProbData & *two*) [inline]

Definition at line 128 of file c2w_array_trie.hpp.

7.18.1.5 bool uva::smt::bpbd::server::lm::__C2WArrayTrie::operator> (const TCtxIdProbData & *one*, const TCtxIdProbData & *two*) [inline]

Definition at line 124 of file c2w_array_trie.hpp.

7.19 uva::smt::bpbd::server::lm::__executor Namespace Reference

Classes

- struct lm_exec_params

7.20 uva::smt::bpbd::server::lm::__G2DMapTrie Namespace Reference

Classes

- struct S_M_GramData

7.21 uva::smt::bpbd::server::lm::__H2DMapTrie Namespace Reference

Classes

- struct S_M_GramData

7.22 uva::smt::bpbd::server::lm::__LayeredTrieBase Namespace Reference

Functions

- template<typename TrieType , phrase_length CURR_LEVEL, bool GET_BACK_OFF_CTX_ID, debug_levels_enum LOG_LEVEL = debug_levels_enum::DEBUG1>
  phrase_length search_m_gram_ctx_id (const TrieType &trie, const word_uid *const word_ids, TLongId &prev_ctx_id, TLongId &ctx_id)
- template<typename TrieType , phrase_length CURR_LEVEL, debug_levels_enum LOG_LEVEL>
  void get_context_id (TrieType &trie, const model_m_gram &gram, TLongId &ctx_id)

7.22.1 Function Documentation

7.22.1.1 template<typename TrieType , phrase_length CURR_LEVEL, debug_levels_enum LOG_LEVEL> void uva::smt::bpbd::server::lm::__LayeredTrieBase::get_context_id (TrieType & *trie,* const model_m_gram & *gram,* TLongId & *ctx_id*) [inline]

This function computes the context id of the N-gram given by the tokens, e.g. [w1 w2 w3 w4]

WARNING: Must be called on M-grams with M > 1!

Parameters

gram	the m-gram we need to compute the context for.
mgram_word_ids	the m-gram word ids aligned to the end of the array
the	resulting hash of the context(w1 w2 w3)

Returns

true if the context was found otherwise false

Definition at line 113 of file layered_trie_base.hpp.

7.22.1.2 template<typename TrieType , phrase_length CURR_LEVEL, bool GET_BACK_OFF_CTX_ID, debug_levels_enum LOG_LEVEL = debug_levels_enum::DEBUG1> phrase_length uva::smt::bpbd::server::lm::__LayeredTrieBase::search_m_gram_ctx_id (const TrieType & *trie,* const word_uid *const *word_ids,* TLongId & *prev_ctx_id,* TLongId & *ctx_id*) [inline]

Allows to obtain the context and previous context id for the sub-m-gram defined by the given template parameters.

Parameters

CURR_LEVEL	the level of the sub-m-gram for which the context id is to be computed
DO_PREV_CONTEXT	true if the previous context id is to be computed, otherwise false

LOG_LEVEL	the desired debug level
word_ids	the array of word ids to consider for computing the context id
prev_ctx_id	the computed previous context id, if computed
ctx_id	the context id, if computed

Returns

the level of the m-gram for which the last context id could be computed

Definition at line 71 of file layered_trie_base.hpp.

7.23 uva::smt::bpbd::server::lm::__W2CArrayTrie Namespace Reference

Classes

- struct S_M_GramData

Typedefs

- typedef S_M_GramData< m_gram_payload > T_M_GramData
- typedef S_M_GramData< prob_weight > T_N_GramData

Functions

- bool operator< (const T_M_GramData &one, const T_M_GramData &two)
- bool operator< (const T_N_GramData &one, const T_N_GramData &two)

7.23.1 Typedef Documentation

7.23.1.1 typedef S_M_GramData<m_gram_payload> uva::smt::bpbd::server::lm::__W2CArrayTrie::T_M_GramData

Definition at line 77 of file w2c_array_trie.hpp.

7.23.1.2 typedef S_M_GramData<prob_weight> uva::smt::bpbd::server::lm::__W2CArrayTrie::T_N_GramData

Definition at line 78 of file w2c_array_trie.hpp.

7.23.2 Function Documentation

7.23.2.1 bool uva::smt::bpbd::server::lm::__W2CArrayTrie::operator< (const T_M_GramData & *one*, const T_M_GramData & *two*) [inline]

This is the less operator implementation

Parameters

one	the first object to compare
two	the second object to compare

Returns

true if ctx_id of one is smaller than ctx_id of two, otherwise false

Definition at line 86 of file w2c_array_trie.hpp.

7.23.2.2 bool uva::smt::bpbd::server::lm::__W2CArrayTrie::operator< (const T_N_GramData & *one*, const T_N_GramData & *two*) [inline]

This is the less operator implementation

Parameters

one	the first object to compare
two	the second object to compare

Returns

true if ctx_id of one is smaller than ctx_id of two, otherwise false

Definition at line 96 of file w2c_array_trie.hpp.

7.24 uva::smt::bpbd::server::lm::__W2CHybridTrie Namespace Reference

7.25 uva::smt::bpbd::server::lm::arpa Namespace Reference

Classes

- class lm_basic_builder
- class lm_gram_builder
- class lm_gram_builder_factory
- struct TAddGramFunct

Functions

- INSTANTIATE_TRIE_BUILDER_FILE_READER (cstyle_file_reader)
- INSTANTIATE_TRIE_BUILDER_FILE_READER (file_stream_reader)
- INSTANTIATE_TRIE_BUILDER_FILE_READER (memory_mapped_file_reader)
- INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_1)
- INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_2)
- INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_3)
- INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_4)
- INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_5)
- INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_6)
- INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_7)

7.25.1 Function Documentation

7.25.1.1 uva::smt::bpbd::server::lm::arpa::INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_1)

7.25.1.2 uva::smt::bpbd::server::lm::arpa::INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_2)

7.25.1.3 uva::smt::bpbd::server::lm::arpa::INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_3)

7.25.1.4 uva::smt::bpbd::server::lm::arpa::INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_4)

7.25.1.5 uva::smt::bpbd::server::lm::arpa::INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_5)

7.25.1.6 uva::smt::bpbd::server::lm::arpa::INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_6)

7.25.1.7 uva::smt::bpbd::server::lm::arpa::INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_7)

7.25.1.8 uva::smt::bpbd::server::lm::arpa::INSTANTIATE_TRIE_BUILDER_FILE_READER (cstyle_file_reader)

7.25.1.9 uva::smt::bpbd::server::lm::arpa::INSTANTIATE_TRIE_BUILDER_FILE_READER (file_stream_reader)

7.25.1.10 uva::smt::bpbd::server::lm::arpa::INSTANTIATE_TRIE_BUILDER_FILE_READER (memory_mapped_file_reader)

7.26 uva::smt::bpbd::server::lm::caching Namespace Reference

Classes

- class BitmapHashCache

7.27 uva::smt::bpbd::server::lm::dictionary Namespace Reference

Namespaces

- __AWordIndex
- __counting_word_index
- __optimizing_word_index

Classes

- class aword_index
- class basic_word_index
- class counting_word_index
- class hashing_word_index
- class optimizing_word_index

Typedefs

- typedef optimizing_word_index< basic_word_index > basic_optimizing_word_index
- typedef optimizing_word_index< counting_word_index > counting_optimizing_word_index

7.27.1 Typedef Documentation

7.27.1.1 typedef optimizing_word_index<basic_word_index> uva::smt::bpbd::server::lm::dictionary::basic_optimizing_word_index

Definition at line 414 of file optimizing_word_index.hpp.

7.27.1.2 typedef optimizing_word_index<counting_word_index> uva::smt::bpbd::server::lm::dictionary::counting_optimizing_word_index

Definition at line 415 of file optimizing_word_index.hpp.

7.28 uva::smt::bpbd::server::lm::dictionary::__AWordIndex Namespace Reference

7.29 uva::smt::bpbd::server::lm::dictionary::__counting_word_index Namespace Reference

Classes

- struct TWordInfo

Functions

- bool operator< (const TWordInfo &one, const TWordInfo &two)

7.29.1 Function Documentation

7.29.1.1 bool uva::smt::bpbd::server::lm::dictionary::__counting_word_index::operator< (const TWordInfo & *one*, const TWordInfo & *two*) [inline]

The comparison operator for two word info objects, the one that is smaller has the highest word probability.

Parameters

one	the first object to compare
two	the second object to compare

Returns

the smaller one is the most used one, with the higher word probability

Definition at line 68 of file counting_word_index.hpp.

7.30 uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index Namespace Reference

Classes

- struct word_index_bucket_entry

Functions

- template<typename word_id_type >
  struct uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index::word_index_bucket_entry __attribute__ ((packed))

7.30.1 Function Documentation

7.30.1.1 template<typename word_id_type > struct uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index::word_index_bucket_entry uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index::__attribute__ ((packed))

7.31 uva::smt::bpbd::server::lm::identifiers Namespace Reference

Typedefs

- typedef uint32_t TShortId
- typedef uint64_t TLongId

7.31.1 Detailed Description

This namespace stores types and constants for the m-gram and context identifiers.

Todo Go trough all the LM and change from TShordId and TLongId to phrase_uid, word_uid and basic types!

7.31.2 Typedef Documentation

7.31.2.1 typedef uint64_t uva::smt::bpbd::server::lm::identifiers::TLongId

Definition at line 75 of file lm_consts.hpp.

7.31.2.2 typedef uint32_t uva::smt::bpbd::server::lm::identifiers::TShortId

Definition at line 73 of file lm_consts.hpp.

7.32 uva::smt::bpbd::server::lm::m_grams Namespace Reference

Namespaces

- m_gram_id

Classes

- struct m_gram_payload_s
- class model_m_gram
- class phrase_base
- class query_m_gram

Typedefs

- typedef m_gram_payload_s m_gram_payload

Functions

- ostream & operator<< (ostream &stream, const model_m_gram &gram)
- ostream & operator<< (ostream &stream, const query_m_gram &gram)

7.32.1 Typedef Documentation

7.32.1.1 typedef m_gram_payload_s uva::smt::bpbd::server::lm::m_grams::m_gram_payload

Definition at line 54 of file m_gram_payload.hpp.

7.32.2 Function Documentation

7.32.2.1 ostream& uva::smt::bpbd::server::lm::m_grams::operator<< (ostream & *stream*, const model_m_gram & *gram*)

Allows to serialize the m-gram to the output stream as a string

Parameters

stream	the reference to the stream to output into
gram	the m-gram object to output

Returns

the reference to the stream

Definition at line 44 of file model_m_gram.cpp.

7.32.2.2 ostream& uva::smt::bpbd::server::lm::m_grams::operator<< (ostream & *stream*, const query_m_gram & *gram*)

Allows to serialize the m-gram to the output stream as a string

Parameters

stream	the reference to the stream to output into
gram	the m-gram object to output

Returns

the reference to the stream

Definition at line 44 of file query_m_gram.cpp.

7.33 uva::smt::bpbd::server::lm::m_grams::m_gram_id Namespace Reference

Classes

- class Byte_M_Gram_Id
- struct T_Gram_Id_Key

Typedefs

- typedef uint8_t * TM_Gram_Id_Value_Ptr

7.33.1 Detailed Description

This namespace stores some generic macros and functions for the m-gram id The ones stored here are not made part of the byte-m-gram-id structure as they are generic and can be used in other m-gram ids.

7.33.2 Typedef Documentation

7.33.2.1 typedef uint8_t* uva::smt::bpbd::server::lm::m_grams::m_gram_id::TM_Gram_Id_Value_Ptr

Definition at line 60 of file m_gram_id.hpp.

7.34 uva::smt::bpbd::server::lm::proxy Namespace Reference

Classes

- class lm_fast_query_proxy
- class lm_fast_query_proxy_local
- class lm_proxy
- class lm_proxy_local
- class lm_slow_query_proxy
- class lm_slow_query_proxy_local

7.35 uva::smt::bpbd::server::rm Namespace Reference

Namespaces

- builders
- models
- proxy

Classes

- class rm_configurator
- struct rm_parameters

Typedefs

- typedef rm_basic_model rm_model_type
- typedef cstyle_file_reader rm_model_reader
- typedef rm_basic_builder< rm_model_type, rm_model_reader > rm_builder_type

7.35.1 Typedef Documentation

7.35.1.1 typedef rm_basic_builder<rm_model_type, rm_model_reader> uva::smt::bpbd::server::rm::rm_builder_type

Definition at line 57 of file rm_configs.hpp.

7.35.1.2 typedef cstyle_file_reader uva::smt::bpbd::server::rm::rm_model_reader

Definition at line 54 of file rm_configs.hpp.

7.35.1.3 typedef rm_basic_model uva::smt::bpbd::server::rm::rm_model_type

Definition at line 51 of file rm_configs.hpp.

7.36 uva::smt::bpbd::server::rm::builders Namespace Reference

Classes

- class rm_basic_builder

7.37 uva::smt::bpbd::server::rm::models Namespace Reference

Namespaces

- __rm_basic_model

Classes

- class rm_basic_model
- class rm_entry_temp
- class rm_query

Typedefs

- typedef rm_entry_temp< NUM_RM_FEATURES > rm_entry

Enumerations

- enum reordering_orientation {
  UNKNOWN_ORIENT = 0, MONOTONE_ORIENT = UNKNOWN_ORIENT + 1, SWAP_ORIENT = MONO↩
  TONE_ORIENT + 1, DISCONT_LEFT_ORIENT = SWAP_ORIENT + 1,
  DISCONT_RIGHT_ORIENT = DISCONT_LEFT_ORIENT + 1, size = DISCONT_RIGHT_ORIENT + 1 }

7.37.1 Typedef Documentation

7.37.1.1 typedef rm_entry_temp<NUM_RM_FEATURES> uva::smt::bpbd::server::rm::models::rm_entry

Definition at line 189 of file rm_entry.hpp.

7.37.2 Enumeration Type Documentation

7.37.2.1 enum uva::smt::bpbd::server::rm::models::reordering_orientation

Defined the reordering orientations in the lexicolized model

Enumerator

UNKNOWN_ORIENT

MONOTONE_ORIENT

SWAP_ORIENT

DISCONT_LEFT_ORIENT

DISCONT_RIGHT_ORIENT

size

Definition at line 53 of file rm_entry.hpp.

7.38 uva::smt::bpbd::server::rm::models::__rm_basic_model Namespace Reference

7.39 uva::smt::bpbd::server::rm::proxy Namespace Reference

Classes

- class rm_proxy
- class rm_proxy_local
- class rm_query_proxy
- class rm_query_proxy_local

7.40 uva::smt::bpbd::server::task_id Namespace Reference

7.41 uva::smt::bpbd::server::tm Namespace Reference

Namespaces

- builders
- models
- proxy

Classes

- class tm_configurator
- struct tm_parameters

Typedefs

- typedef tm_basic_model tm_model_type
- typedef cstyle_file_reader tm_model_reader
- typedef tm_basic_builder< tm_model_type, tm_model_reader > tm_builder_type

7.41.1 Typedef Documentation

7.41.1.1 typedef tm_basic_builder<tm_model_type, tm_model_reader> uva::smt::bpbd::server::tm::tm_builder_type

Definition at line 57 of file tm_configs.hpp.

7.41.1.2 typedef cstyle_file_reader uva::smt::bpbd::server::tm::tm_model_reader

Definition at line 54 of file tm_configs.hpp.

7.41.1.3 typedef tm_basic_model uva::smt::bpbd::server::tm::tm_model_type

Definition at line 51 of file tm_configs.hpp.

7.42 uva::smt::bpbd::server::tm::builders Namespace Reference

Classes

- class tm_basic_builder

Typedefs

- typedef unordered_map< phrase_uid, size_t > sizes_map

7.42.1 Typedef Documentation

7.42.1.1 typedef unordered_map<phrase_uid, size_t> uva::smt::bpbd::server::tm::builders::sizes_map

Definition at line 73 of file tm_basic_builder.hpp.

7.43 uva::smt::bpbd::server::tm::models Namespace Reference

Namespaces

- __tm_basic_model

Classes

- class tm_basic_model
- class tm_query
- class tm_source_entry
- class tm_target_entry_temp

Typedefs

- typedef const tm_source_entry tm_const_source_entry
- typedef tm_const_source_entry * tm_const_source_entry_ptr
- typedef tm_source_entry * tm_source_entry_ptr
- typedef tm_target_entry_temp< NUM_TM_FEATURES > tm_target_entry
- typedef const tm_target_entry tm_const_target_entry
- typedef prob_weight feature_array[tm_target_entry::NUM_FEATURES]

7.43.1 Typedef Documentation

7.43.1.1 typedef prob_weight uva::smt::bpbd::server::tm::models::feature_array[tm_target_entry::NUM_FEATURES]

Definition at line 244 of file tm_target_entry.hpp.

7.43.1.2 typedef const tm_source_entry uva::smt::bpbd::server::tm::models::tm_const_source_entry

Definition at line 270 of file tm_source_entry.hpp.

7.43.1.3 typedef tm_const_source_entry* uva::smt::bpbd::server::tm::models::tm_const_source_entry_ptr

Definition at line 273 of file tm_source_entry.hpp.

7.43.1.4 typedef const tm_target_entry uva::smt::bpbd::server::tm::models::tm_const_target_entry

Definition at line 241 of file tm_target_entry.hpp.

7.43.1.5 typedef tm_source_entry* uva::smt::bpbd::server::tm::models::tm_source_entry_ptr

Definition at line 276 of file tm_source_entry.hpp.

7.43.1.6 typedef tm_target_entry_temp<NUM_TM_FEATURES> uva::smt::bpbd::server::tm::models::tm_↩ target_entry

Definition at line 238 of file tm_target_entry.hpp.

7.44 uva::smt::bpbd::server::tm::models::__tm_basic_model Namespace Reference

7.45 uva::smt::bpbd::server::tm::proxy Namespace Reference

Classes

- class tm_proxy
- class tm_proxy_local
- class tm_query_proxy
- class tm_query_proxy_local

7.46 uva::utils Namespace Reference

Namespaces

- containers
- exceptions
- file
- hashing
- logging
- math
- monitor

- text
- threads

7.47 uva::utils::containers Namespace Reference

Namespaces

- alloc
- utils

Classes

- class circular_queue
- class dynamic_stack_array
- struct ELEMENT_DEALLOC_FUNC
- class fixed_size_hashmap
- class greedy_memory_storage
- class mem_increase_strategy
- class upp_diag_matrix

Typedefs

- typedef std::function< size_t(const size_t) > TCapacityIncFunct

Enumerations

- enum mem_inc_types_enum {
  UNDEFINED = 0, CONSTANT = UNDEFINED + 1, LINEAR = CONSTANT + 1, LOG_2 = LINEAR + 1,
  LOG_10 = LOG_2 + 1, size = LOG_10 + 1 }

Functions

- mem_increase_strategy get_mem_incr_strat (const mem_inc_types_enum stype, const size_t min_mem_↩
  inc, const size_t mem_inc_factor)

Variables

- const char *const _memIncTypesEnumStr [mem_inc_types_enum::size] = {"CONSTANT", "LINEAR", "LO↩
  G_2", "LOG_10"}

7.47.1 Typedef Documentation

7.47.1.1 typedef std::function<size_t(const size_t) > uva::utils::containers::TCapacityIncFunct

This is a function type for the function that should compute the capacity increase

Parameters

the	first argument is the current capacity as float

Returns

the capacity increase

Definition at line 63 of file dynamic_memory_arrays.hpp.

7.47.2 Enumeration Type Documentation

7.47.2.1 enum uva::utils::containers::mem_inc_types_enum

Stores the possible memory increase types

Enumerator

UNDEFINED

CONSTANT

LINEAR

LOG_2

LOG_10

size

Definition at line 49 of file dynamic_memory_arrays.hpp.

7.47.3 Function Documentation

7.47.3.1 mem_increase_strategy uva::utils::containers::get_mem_incr_strat (const mem_inc_types_enum *stype,* const size_t *min_mem_inc,* const size_t *mem_inc_factor*) [inline]

This is a factory function allowing to ge the strategy object for the given parameters.

Todo Optimize the switch, it is pretty ugly, use a map or something.

Parameters

stype	the strategy type
min_mem_inc	the minimum memory increment in number of elements
mem_inc_factor	the memory increment factor, the number we will multiply by the computed increment

Returns

the pointer to a newly allocated strategy object

Definition at line 147 of file dynamic_memory_arrays.hpp.

7.47.4 Variable Documentation

7.47.4.1 const char* const uva::utils::containers::_memIncTypesEnumStr[mem_inc_types_enum::size] = {"CONSTANT", "LINEAR", "LOG_2", "LOG_10"}

Stores the string names of the memory increase strategies, should correspond with the enum MemIncTypesEnum indexes!

Definition at line 69 of file dynamic_memory_arrays.hpp.

7.48 uva::utils::containers::alloc Namespace Reference

Classes

- class greedy_memory_allocator

Functions

- template<typename TContaner , typename TAllocator >
  void allocate_container (TContaner **ppContainer, TAllocator **ppAllocator, const size_t numEntries, const string ctName, const float factor=UNORDERED_MAP_MEMORY_FACTOR)
- template<typename TContaner , typename TAllocator >
  void reserve_mem_unordered_map (TContaner **ppContainer, TAllocator **ppAllocator, const size_t numEntries, const string ctName, const float factor=UNORDERED_MAP_MEMORY_FACTOR)
- template<typename TContaner , typename TAllocator >
  void deallocate_container (TContaner **ppContainer, TAllocator **ppAllocator)
- template<typename T , typename U >
  bool operator== (const greedy_memory_allocator< T > &, const greedy_memory_allocator< U > &)
- template<typename T >
  bool operator== (const greedy_memory_allocator< T > &, const greedy_memory_allocator< T > &)
- template<typename T , typename U >
  bool operator!= (const greedy_memory_allocator< T > &, const greedy_memory_allocator< U > &)
- template<typename T >
  bool operator!= (const greedy_memory_allocator< T > &, const greedy_memory_allocator< T > &)

7.48.1 Function Documentation

7.48.1.1 template<typename TContaner , typename TAllocator > void uva::utils::containers::alloc::allocate_container (TContaner ** *ppContainer,* TAllocator ** *ppAllocator,* const size_t *numEntries,* const string *ctName,* const float *factor* = UNORDERED_MAP_MEMORY_FACTOR)

This is helper function that allows to allocate the container, allocator and the actual data storage Note that, this functions is meant to be used with the unordered_map allocator

Parameters

ppContainer	the pointer to the container pointer
ppAllocator	the pointer to the allocator pointer
numEntries	the number of entries to pre-allocate for
ctName	the container name for logging purposes
factor	the memory multiplication factor, default is UNORDERED_MAP_MEMORY_FACTOR. This is how many times memory we will allocate (than needed to store numEntries elems)

Definition at line 65 of file greedy_memory_allocator.hpp.

7.48.1.2 template<typename TContaner , typename TAllocator > void uva::utils::containers::alloc::deallocate_container (TContaner ** *ppContainer,* TAllocator ** *ppAllocator*)

This is helper function that allows to deallocate the container allocator and actual data storage

Parameters

ppContainer	the pointer to the container pointer

ppAllocator	the pointer to the allocator pointer
ppStorage	the pointer to the storage pointer

Definition at line 113 of file greedy_memory_allocator.hpp.

7.48.1.3 template<typename T , typename U > bool uva::utils::containers::alloc::operator!= (const greedy_memory_allocator< T > & , const greedy_memory_allocator< U > &)

Definition at line 311 of file greedy_memory_allocator.hpp.

7.48.1.4 template<typename T > bool uva::utils::containers::alloc::operator!= (const greedy_memory_allocator< T > & , const greedy_memory_allocator< T > &)

Definition at line 316 of file greedy_memory_allocator.hpp.

7.48.1.5 template<typename T , typename U > bool uva::utils::containers::alloc::operator== (const greedy_memory_allocator< T > & , const greedy_memory_allocator< U > &)

Definition at line 301 of file greedy_memory_allocator.hpp.

7.48.1.6 template<typename T > bool uva::utils::containers::alloc::operator== (const greedy_memory_allocator< T > & , const greedy_memory_allocator< T > &)

Definition at line 306 of file greedy_memory_allocator.hpp.

7.48.1.7 template<typename TContaner , typename TAllocator > void uva::utils::containers::alloc::reserve_mem_unordered_map (TContaner ** *ppContainer,* TAllocator ** *ppAllocator,* const size_t *numEntries,* const string *ctName,* const float *factor* = UNORDERED_MAP_MEMORY_FACTOR)

This is helper function that allows to allocate the container, allocator and the actual data storage Note that, this functions is meant to be used with the unordered_map allocator

Parameters

ppContainer	the pointer to the container pointer
ppAllocator	the pointer to the allocator pointer
numEntries	the number of entries to pre-allocate for
ctName	the container name for logging purposes
factor	the memory multiplication factor, default is UNORDERED_MAP_MEMORY_FACTOR. This is how many times memory we will allocate (than needed to store numEntries elems)

Definition at line 95 of file greedy_memory_allocator.hpp.

7.49 uva::utils::containers::utils Namespace Reference

Classes

- struct T_IS_COMPARE_FUNC

Functions

- template<typename ARR_ELEM_TYPE >

bool my_bsearch_id (const ARR_ELEM_TYPE *array, int64_t l_idx, int64_t u_idx, const typename ARR_↩ ELEM_TYPE::TIdType key, const ARR_ELEM_TYPE *&found_elem)

- template<typename ARR_ELEM_TYPE , typename IDX_TYPE , typename KEY_TYPE >
  bool my_bsearch_id (const ARR_ELEM_TYPE *array, int64_t l_idx, int64_t u_idx, const KEY_TYPE key, IDX_TYPE &found_pos)
- template<typename ARR_ELEM_TYPE >
  bool my_bsearch_wordId_ctxId (const ARR_ELEM_TYPE *array, int64_t l_idx, int64_t u_idx, const uint32_t key1, const uint32_t key2, uint32_t &found_pos)
- template<typename ARR_ELEM_TYPE , typename KEY_TYPE >
  bool my_isearch_id (const ARR_ELEM_TYPE *array, int64_t l_idx, int64_t u_idx, const KEY_TYPE key, const ARR_ELEM_TYPE *&found_elem)
- template<typename ARR_ELEM_TYPE , typename INDEX_TYPE , typename KEY_TYPE >
  bool my_bsearch (const ARR_ELEM_TYPE *array, INDEX_TYPE l_idx, INDEX_TYPE u_idx, const KEY_↩ TYPE key, INDEX_TYPE &mid_pos)
- template<typename ARR_ELEM_TYPE >
  bool my_lsearch_id (const ARR_ELEM_TYPE *array, int64_t l_idx, int64_t u_idx, const typename ARR_E↩ LEM_TYPE::TIdType key, const ARR_ELEM_TYPE *&found_elem)
- template<typename ELEM_TYPE >
  void my_sort (ELEM_TYPE *array_begin, const uint32_t array_size, typename T_IS_COMPARE_FUNC< ELEM_TYPE >::func_type is_less_func)
- template<typename ELEM_TYPE , typename T_IS_COMPARE_FUNC< ELEM_TYPE >::func_ptr IS_LESS_FUNC>
  void my_sort (ELEM_TYPE *array_begin, const uint32_t array_size)
- template<typename ELEM_TYPE , bool IS_PROGRESS = true>
  bool is_less (const ELEM_TYPE &first, const ELEM_TYPE &second)

7.49.1 Function Documentation

7.49.1.1 template<typename ELEM_TYPE , bool IS_PROGRESS = true> bool uva::utils::containers::utils::is_less (const ELEM_TYPE & *first,* const ELEM_TYPE & *second*) [inline]

The basic "is less" function for the sort algorithms that allows to update the progress bar.

Parameters

first	the first element to compare
second	the second element to compare

Returns

true if the first element is less then the second

Definition at line 429 of file array_utils.hpp.

7.49.1.2 template<typename ARR_ELEM_TYPE , typename INDEX_TYPE , typename KEY_TYPE > bool uva::utils::containers::utils::my_bsearch (const ARR_ELEM_TYPE * *array,* INDEX_TYPE *l_idx,* INDEX_TYPE *u_idx,* const KEY_TYPE *key,* INDEX_TYPE & *mid_pos*)

This is a search algorithm for some ordered array, here we use bsearch from <cstdlib>

Parameters

array	the pointer to the first array element
l_idx	the initial left border index for searching

u_idx	the initial right border index for searching
key	the key we are searching for
mid_pos	the out parameter that stores the found element index, if any

Returns

true if the element was found, otherwise false

Exceptions

Exception	in case (l_idx $<$ 0) $||$ (l_idx $>$ u_idx), with sanity checks on

Definition at line 295 of file array_utils.hpp.

7.49.1.3 template<typename ARR_ELEM_TYPE > bool uva::utils::containers::utils::my_bsearch_id (const ARR_ELEM_TYPE * *array,* int64_t *l_idx,* int64_t *u_idx,* const typename ARR_ELEM_TYPE::TIdType *key,* const ARR_ELEM_TYPE *& *found_elem*) [inline]

This is a binary search algorithm for some ordered array

Parameters

ARR_ELEM_T↩ YPE	the array element structure, must have ctx_id field as this method will specifically use it to compare elements.
IDX_TYPE	the index type
KEY_TYPE	the key type template parameter
array	the pointer to the first array element
l_idx	the initial left border index for searching
u_idx	the initial right border index for searching
key	the key we are searching for
found_pos	the out parameter that stores the found element index, if any

Returns

true if the element was found, otherwise false

Exceptions

Exception	in case (l_idx $<$ 0) $||$ (l_idx $>$ u_idx), with sanity checks on

Definition at line 180 of file array_utils.hpp.

7.49.1.4 template<typename ARR_ELEM_TYPE , typename IDX_TYPE , typename KEY_TYPE > bool uva::utils::containers::utils::my_bsearch_id (const ARR_ELEM_TYPE * *array,* int64_t *l_idx,* int64_t *u_idx,* const KEY_TYPE *key,* IDX_TYPE & *found_pos*) [inline]

This is a binary search algorithm for some ordered array

Parameters

ARR_ELEM_T↩ YPE	the array element structure, must have ctx_id field as this method will specifically use it to compare elements.
IDX_TYPE	the index type
KEY_TYPE	the key type template parameter

array	the pointer to the first array element
l_idx	the initial left border index for searching
u_idx	the initial right border index for searching
key	the key we are searching for
found_pos	the out parameter that stores the found element index, if any

Returns

true if the element was found, otherwise false

Exceptions

Exception	in case (l_idx $<$ 0) $||$ (l_idx $>$ u_idx), with sanity checks on

Definition at line 200 of file array_utils.hpp.

7.49.1.5 template<typename ARR_ELEM_TYPE > bool uva::utils::containers::utils::my_bsearch_wordId_ctxId (const ARR_ELEM_TYPE * *array*, int64_t *l_idx*, int64_t *u_idx*, const uint32_t *key1*, const uint32_t *key2*, uint32_t & *found_pos*) [inline]

This is a binary search algorithm for some ordered array for two keys

Parameters

ARR_ELEM_T↩ YPE	the array element structure, must have word_id field as this method will specifically use it to compare elements.
array	the pointer to the first array element
l_idx	the initial left border index for searching
u_idx	the initial right border index for searching
key	the key we are searching for
found_pos	the out parameter that stores the found element index, if any

Returns

true if the element was found, otherwise false

Exceptions

Exception	in case (l_idx $<$ 0) $||$ (l_idx $>$ u_idx), with sanity checks on

Definition at line 216 of file array_utils.hpp.

7.49.1.6 template<typename ARR_ELEM_TYPE , typename KEY_TYPE > bool uva::utils::containers::utils::my_isearch_id (const ARR_ELEM_TYPE * *array*, int64_t *l_idx*, int64_t *u_idx*, const KEY_TYPE *key*, const ARR_ELEM_TYPE *& *found_elem*)

This is an interpolated search algorithm for some ordered array WARNING: IS ACTUALLY VERT SLOW at least in the current implementation and for the current application!

Parameters

ARR_ELEM_T↩ YPE	the array element structure, must have ctx_id field as this method will specifically use it to compare elements.
IDX_TYPE	the index type

KEY_TYPE	the key type template parameter
array	the pointer to the first array element
l_idx	the initial left border index for searching
u_idx	the initial right border index for searching
key	the key we are searching for
found_pos	the out parameter that stores the found element index, if any

Returns

true if the element was found, otherwise false

Exceptions

Exception	in case (l_idx $<$ 0) $||$ (l_idx $>$ u_idx), with sanity checks on

Definition at line 235 of file array_utils.hpp.

7.49.1.7 template<typename ARR_ELEM_TYPE > bool uva::utils::containers::utils::my_lsearch_id (const ARR_ELEM_TYPE * *array,* int64_t *l_idx,* int64_t *u_idx,* const typename ARR_ELEM_TYPE::TIdType *key,* const ARR_ELEM_TYPE *& *found_elem*) [inline]

This is a linear search algorithm for some ordered array

Parameters

ARR_ELEM_TYPE	the array element structure
IDX_TYPE	the index type
KEY_TYPE	the key type template parameter
array	the pointer to the first array element
l_idx	the initial left border index for searching
u_idx	the initial right border index for searching
key	the key we are searching for
found_pos	the out parameter that stores the found element index, if any

Returns

true if the element was found, otherwise false

Exceptions

Exception	in case (l_idx $<$ 0) $||$ (l_idx $>$ u_idx), with sanity checks on

Definition at line 355 of file array_utils.hpp.

7.49.1.8 template<typename ELEM_TYPE > void uva::utils::containers::utils::my_sort (ELEM_TYPE * *array_begin,* const uint32_t *array_size,* typename T_IS_COMPARE_FUNC< ELEM_TYPE >::func_type *is_less_func*) [inline]

This methos is used to do <algorithm> std::sort on an array of structures convertable to some simple comparable type. This method does the progress bar update, if needed

Parameters

ELEM_TYPE	the array element type

array_begin	the pointer to the array's first element
array_size	the size of the array
is_less_func	the is-less function

Definition at line 395 of file array_utils.hpp.

7.49.1.9 template<typename ELEM_TYPE , typename T_IS_COMPARE_FUNC< ELEM_TYPE >::func_ptr IS_LESS_FUNC> void uva::utils::containers::utils::my_sort (ELEM_TYPE * *array_begin*, const uint32_t *array_size*) [inline]

This methos is used to do <algorithm> std::sort on an array of structures convertable to some simple comparable type. This method does the progress bar update, if needed

Parameters

ELEM_TYPE	the array element type
IS_LESS_FU↩ NC	the is-less function
array_begin	the pointer to the array's first element
array_size	the size of the array

This methos is used to do <algorithm> std::sort on an array of structures convertable to some simple comparable type. This method does the progress bar update, if needed

Parameters

ELEM_TYPE	the array element type
IS_PROGRESS	if true the progress bar will be updated, otherwise not, default is true
array_begin	the pointer to the array's first element
array_size	the size of the array

Definition at line 414 of file array_utils.hpp.

7.50 uva::utils::exceptions Namespace Reference

Classes

- class uva_exception

Variables

- constexpr bool DO_SANITY_CHECKS = false

7.50.1 Variable Documentation

7.50.1.1 constexpr bool uva::utils::exceptions::DO_SANITY_CHECKS = false

Definition at line 42 of file exceptions.hpp.

7.51 uva::utils::file Namespace Reference

Classes

- class afile_reader
- class cstyle_file_reader
- class file_stream_reader

- class memory_mapped_file_reader
- class text_piece_reader

Functions

- ostream & operator<< (ostream &output, const text_piece_reader &val)
- template<size_t NUM_TOKENS>
  string tokens_to_string (const text_piece_reader tokens[NUM_TOKENS], const size_t begin_idx, const size_t end_idx)

7.51.1 Function Documentation

7.51.1.1 ostream& uva::utils::file::operator<< (ostream & *output,* const text_piece_reader & *val*) [inline]

Overloading the output operator for the ostream

Parameters

output	the stream to print to
val	the value to print

Returns

the output stream

Definition at line 561 of file text_piece_reader.hpp.

7.51.1.2 template<size_t NUM_TOKENS> string uva::utils::file::tokens_to_string (const text_piece_reader *tokens[NUM_TOKENS],* const size_t *begin_idx,* const size_t *end_idx*) [inline]

This function allows to convert the BasicTextFileReader elements tokens into a array string representation.

Parameters

tokens	the tokens to print
from_idx	the from index
to_idx	the to index

Returns

the resulting string

Definition at line 573 of file text_piece_reader.hpp.

7.52 uva::utils::hashing Namespace Reference

7.53 uva::utils::logging Namespace Reference

Classes

- class logger
- struct logging_synch

Enumerations

- enum debug_levels_enum {
  ERROR = 0, WARNING = ERROR + 1, USAGE = WARNING + 1, RESULT = USAGE + 1,
  INFO = RESULT + 1, INFO1 = INFO + 1, INFO2 = INFO1 + 1, INFO3 = INFO2 + 1,
  DEBUG = INFO3 + 1, DEBUG1 = DEBUG + 1, DEBUG2 = DEBUG1 + 1, DEBUG3 = DEBUG2 + 1,
  DEBUG4 = DEBUG3 + 1, size = DEBUG4 + 1 }

Functions

- std::ostream & operator<< (std::ostream &stream, const unsigned char &value)
- std::ostream & operator<< (std::ostream &stream, const signed char &value)

7.53.1 Enumeration Type Documentation

7.53.1.1 enum uva::utils::logging::debug_levels_enum

This enumeration stores all the available logging levels.

Enumerator

ERROR

WARNING

USAGE

RESULT

INFO

INFO1

INFO2

INFO3

DEBUG

DEBUG1

DEBUG2

DEBUG3

DEBUG4

size

Definition at line 48 of file logger.hpp.

7.53.2 Function Documentation

7.53.2.1 std::ostream & uva::utils::logging::operator<< (std::ostream & *stream,* const unsigned char & *value*)

Definition at line 42 of file logger.cpp.

7.53.2.2 std::ostream & uva::utils::logging::operator<< (std::ostream & *stream,* const signed char & *value*)

Definition at line 46 of file logger.cpp.

7.54 uva::utils::math Namespace Reference

Namespaces

- bits
- const_expr
- log2

7.55 uva::utils::math::bits Namespace Reference

7.56 uva::utils::math::const_expr Namespace Reference

Functions

- constexpr double log2 (double value, double pow=0.0)
- constexpr uint64_t ceil (double value)
- constexpr uint64_t power (uint64_t value, uint8_t pow)

7.56.1 Function Documentation

7.56.1.1 constexpr uint64_t uva::utils::math::const_expr::ceil (double *value*) [inline]

Definition at line 66 of file math_utils.hpp.

7.56.1.2 constexpr double uva::utils::math::const_expr::log2 (double *value*, double *pow* = 0.0) [inline]

This is a limited implementation of log, the argument value must be >= 1. The computations are also not exact, if the value of the logarithm is not a natural number then we return the maximum integer smaller than the log value plus 0.5. Also if the value is <= 1.0 then the result is 0.0

Definition at line 62 of file math_utils.hpp.

7.56.1.3 constexpr uint64_t uva::utils::math::const_expr::power (uint64_t *value*, uint8_t *pow*) [inline]

Definition at line 72 of file math_utils.hpp.

7.57 uva::utils::math::log2 Namespace Reference

7.58 uva::utils::monitor Namespace Reference

Classes

- struct memory_usage
- class stat_monitor

Typedefs

- typedef memory_usage TMemotyUsage

Variables

- const uint32_t BYTES_ONE_MB = 1024u

7.58.1 Typedef Documentation

7.58.1.1 typedef memory_usage uva::utils::monitor::TMemotyUsage

Definition at line 60 of file statistics_monitor.hpp.

7.58.2 Variable Documentation

7.58.2.1 const uint32_t uva::utils::monitor::BYTES_ONE_MB = 1024u

Definition at line 99 of file statistics_monitor.hpp.

7.59 uva::utils::text Namespace Reference

Variables

- const string UTF8_ASCII_WHITESPACES = u8"\t\f\v\n\r "
- const string UTF8_ASCII_PUNCTUATIONS = u8".,?!/'\"‘@#$%^&∗()[ ]{}-_+=∗<>∼|\\;:"
- const char ASCII_SPACE_CHAR = ' '
- const string UTF8_SPACE_STRING = u8" "
- const string UTF8_EMPTY_STRING = u8""
- const string UTF8_NEW_LINE_STRING = u8"\n"

7.59.1 Variable Documentation

7.59.1.1 const char uva::utils::text::ASCII_SPACE_CHAR = ' '

Definition at line 87 of file string_utils.hpp.

7.59.1.2 const string uva::utils::text::UTF8_ASCII_PUNCTUATIONS = u8".,?!/'\"‘@#$%^&∗()[ ]{}-_+=∗<>∼|\\;:"

Definition at line 85 of file string_utils.hpp.

7.59.1.3 const string uva::utils::text::UTF8_ASCII_WHITESPACES = u8"\t\f\v\n\r "

Definition at line 83 of file string_utils.hpp.

7.59.1.4 const string uva::utils::text::UTF8_EMPTY_STRING = u8""

Definition at line 91 of file string_utils.hpp.

7.59.1.5 const string uva::utils::text::UTF8_NEW_LINE_STRING = u8"\n"

Definition at line 93 of file string_utils.hpp.

7.59.1.6 const string uva::utils::text::UTF8_SPACE_STRING = u8" "

Definition at line 89 of file string_utils.hpp.

7.60 uva::utils::threads Namespace Reference

Typedefs

- typedef lock_guard< recursive_mutex > recursive_guard
- typedef lock_guard< mutex > scoped_guard
- typedef unique_lock< mutex > unique_guard
- typedef atomic< bool > a_bool_flag
- typedef const a_bool_flag & acr_bool_flag

7.60.1 Typedef Documentation

7.60.1.1 typedef atomic<bool> uva::utils::threads::a_bool_flag

Definition at line 51 of file threads.hpp.

7.60.1.2 typedef const a_bool_flag& uva::utils::threads::acr_bool_flag

Definition at line 54 of file threads.hpp.

7.60.1.3 typedef lock_guard<recursive_mutex> uva::utils::threads::recursive_guard

Definition at line 42 of file threads.hpp.

7.60.1.4 typedef lock_guard<mutex> uva::utils::threads::scoped_guard

Definition at line 45 of file threads.hpp.

7.60.1.5 typedef unique_lock<mutex> uva::utils::threads::unique_guard

Definition at line 48 of file threads.hpp.

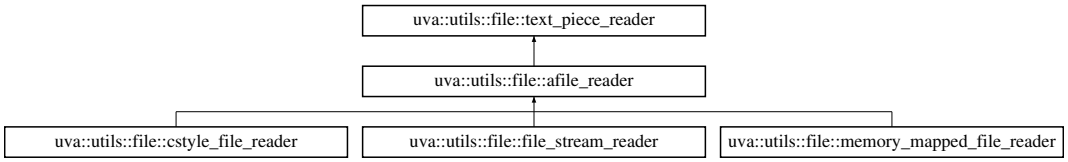
Chapter 8

Class Documentation

8.1 uva::utils::file::afile_reader Class Reference

`#include <afile_reader.hpp>`

Inheritance diagram for uva::utils::file::afile_reader:



Public Member Functions

- afile_reader ()
- virtual bool is_open () const =0
- virtual operator bool () const =0
- virtual void log_reader_type_info ()=0
- virtual void reset ()
- template<const char delim, const uint8_t delim_len = 1>
  bool get_first (text_piece_reader &out)
- template<const char delim, const uint8_t delim_len = 1>
  bool get_last (text_piece_reader &out)
- bool get_first_line (text_piece_reader &out)
- bool get_first_space (text_piece_reader &out)
- bool get_last_space (text_piece_reader &out)
- bool get_first_tab (text_piece_reader &out)
- virtual void close ()
- virtual ~afile_reader ()

Additional Inherited Members

8.1.1 Detailed Description

This is an abstract base class for the file readers we are going to be using to read model files

Definition at line 45 of file afile_reader.hpp.

8.1.2 Constructor & Destructor Documentation

8.1.2.1 uva::utils::file::afile_reader::afile_reader ( ) [inline]

Definition at line 48 of file afile_reader.hpp.

8.1.2.2 virtual uva::utils::file::afile_reader::~afile_reader ( ) [inline], [virtual]

The basic destructor, calls the close method

Definition at line 136 of file afile_reader.hpp.

8.1.3 Member Function Documentation

8.1.3.1 virtual void uva::utils::file::afile_reader::close ( ) [inline], [virtual]

Allows to close the file

Reimplemented in uva::utils::file::memory_mapped_file_reader, uva::utils::file::cstyle_file_reader, and uva::utils::file::file_stream_reader.

Definition at line 130 of file afile_reader.hpp.

8.1.3.2 template<const char delim, const uint8_t delim_len = 1> bool uva::utils::file::afile_reader::get_first (text_piece_reader & *out*) [inline]

This function searches forward for the first occurrence of the argument delimiter symbol.

Definition at line 82 of file afile_reader.hpp.

8.1.3.3 bool uva::utils::file::afile_reader::get_first_line (text_piece_reader & *out*) [inline]

Each file reader implementation will need to override these method, if needed. The method is non-virtual for performance reasons!

Definition at line 99 of file afile_reader.hpp.

8.1.3.4 bool uva::utils::file::afile_reader::get_first_space (text_piece_reader & *out*) [inline]

Each file reader implementation will need to override these method, if needed. The method is non-virtual for performance reasons!

Definition at line 107 of file afile_reader.hpp.

8.1.3.5 bool uva::utils::file::afile_reader::get_first_tab (text_piece_reader & *out*) [inline]

Each file reader implementation will need to override these method, if needed. The method is non-virtual for performance reasons!

Definition at line 123 of file afile_reader.hpp.

8.1.3.6 template<const char delim, const uint8_t delim_len = 1> bool uva::utils::file::afile_reader::get_last (text_piece_reader & *out*) [inline]

This function searches backwards for the first occurrence of the argument delimiter symbol.

Definition at line 91 of file afile_reader.hpp.

8.1.3.7 bool uva::utils::file::afile_reader::get_last_space (text_piece_reader & *out*) [inline]

Each file reader implementation will need to override these method, if needed. The method is non-virtual for performance reasons!

Definition at line 115 of file afile_reader.hpp.

8.1.3.8 virtual bool uva::utils::file::afile_reader::is_open ( ) const [pure virtual]

Allows to check if the file has been open

Returns

true if the file is open otherwise false

Implemented in uva::utils::file::memory_mapped_file_reader, uva::utils::file::cstyle_file_reader, and uva::utils::file::file_stream_reader.

8.1.3.9 virtual void uva::utils::file::afile_reader::log_reader_type_info ( ) [pure virtual]

Allows to log the information about the instantiated file reader type

Implemented in uva::utils::file::memory_mapped_file_reader, uva::utils::file::cstyle_file_reader, and uva::utils::file::file_stream_reader.

8.1.3.10 virtual uva::utils::file::afile_reader::operator bool ( ) const [pure virtual]

Allows to convert the file reader to a boolean

Returns

true if the file exists, otherwise false

Implemented in uva::utils::file::memory_mapped_file_reader, uva::utils::file::cstyle_file_reader, and uva::utils::file::file_stream_reader.

8.1.3.11 virtual void uva::utils::file::afile_reader::reset ( ) [inline], [virtual]

This method allows to reset the reading process and start reading the file from th first line again. The default implementation throws an exception.

Reimplemented in uva::utils::file::cstyle_file_reader, and uva::utils::file::file_stream_reader.

Definition at line 73 of file afile_reader.hpp.

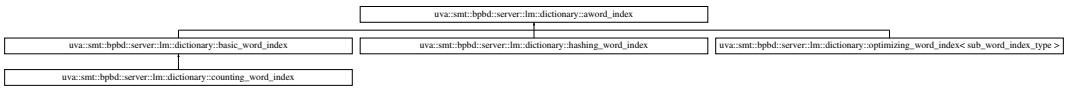
The documentation for this class was generated from the following file:

- inc/common/utils/file/afile_reader.hpp

8.2 uva::smt::bpbd::server::lm::dictionary::aword_index Class Reference

`#include <aword_index.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::dictionary::aword_index:



Public Member Functions

- void reserve (const size_t num_words)
- size_t get_number_of_words (const size_t num_words) const
- word_uid get_word_id (const text_piece_reader &token) const
- bool is_word_registering_needed () const
- word_uid register_word (const text_piece_reader &token)
- bool is_word_counts_needed () const
- void count_word (const text_piece_reader &word, prob_weight prob)
- void do_post_word_count ()
- bool is_post_actions_needed () const
- void do_post_actions ()
- virtual ∼aword_index ()

Static Public Member Functions

- static constexpr bool is_word_index_continuous ()

8.2.1 Detailed Description

This abstract class is used to represent the word dictionary. It contains no specific implementation but is more of an interface. It is used to allow for more word dictionary/index implementations.

Any implementation of this class must issue the unknown word <unk> index 1 (UNKNOWN_WORD_ID).

The first real word index will be therefore 2 (MIN_KNOWN_WORD_ID).

The issued word ids must be continuous and non-repeating, unique!

NOTE: All of the methods are non-virtual for the sake of avoiding virtual method call overheads!

Definition at line 60 of file aword_index.hpp.

8.2.2 Constructor & Destructor Documentation

8.2.2.1 virtual uva::smt::bpbd::server::lm::dictionary::aword_index::∼aword_index ( ) [inline], [virtual]

The basic destructor

Definition at line 178 of file aword_index.hpp.

8.2.3 Member Function Documentation

8.2.3.1 void uva::smt::bpbd::server::lm::dictionary::aword_index::count_word (const text_piece_reader & *word*, prob_weight *prob*) [inline]

This method is to be used when the word counting is needed. The main application here is to first count the number of word usages and then distribute the word ids in such a way that the most used words get the lowest ids.

Parameters

token	the word to count
prob	the word log probability

Definition at line 128 of file aword_index.hpp.

8.2.3.2 void uva::smt::bpbd::server::lm::dictionary::aword_index::do_post_actions ( ) [inline]

Is to be called if the post actions are needed right after that all the individual words have beed added into the index.

Definition at line 154 of file aword_index.hpp.

8.2.3.3 void uva::smt::bpbd::server::lm::dictionary::aword_index::do_post_word_count ( ) [inline]

Should be called if the word count is needed after all the words have been counted.

Definition at line 136 of file aword_index.hpp.

8.2.3.4 size_t uva::smt::bpbd::server::lm::dictionary::aword_index::get_number_of_words (const size_t *num_words*) const [inline]

Allows to get the total words count including the unknown and undefined words

Parameters

num_words	the number of words in the language model

Definition at line 75 of file aword_index.hpp.

8.2.3.5 word_uid uva::smt::bpbd::server::lm::dictionary::aword_index::get_word_id (const text_piece_reader & *token*) const [inline]

This function gets an id for the given word word based no the stored 1-Grams. Continuous word index: If the word is not known then an unknown word ID is returned: UNKNOWN_WORD_ID Discontinuous word index: The returned word id is always >= MIN_KNOWN_WORD_ID

Parameters

token	the word to hash

Returns

the word id or UNKNOWN_WORD_ID if the word is not found

Definition at line 87 of file aword_index.hpp.

8.2.3.6 bool uva::smt::bpbd::server::lm::dictionary::aword_index::is_post_actions_needed ( ) const [inline]

Indicates if the post-actions are needed. The post actions should be called after all the words have been filled into the index.

Returns

true if the post-actions are needed, otherwise false

Definition at line 146 of file aword_index.hpp.

8.2.3.7 bool uva::smt::bpbd::server::lm::dictionary::aword_index::is_word_counts_needed ( ) const [inline]

This method allows to indicate whether word counting is needed by the given implementation of the word index.

Returns

true if the word counting is needed, otherwise false.

Definition at line 116 of file aword_index.hpp.

8.2.3.8 static constexpr bool uva::smt::bpbd::server::lm::dictionary::aword_index::is_word_index_continuous ( ) [inline], [static]

Allows to indicate if the word index is continuous, i.e. it issues the word ids in a continuous range starting from 0. Where 0 and 1 are reserved word ids.

If the word index is not continuous then the uni-gram payload can not be stored in a word id indexed array. Moreover, any word id considered to be a known word, i.e. the unknown word id is never returned by the word index.

This method is to be overridden by the children classes. The default implementation returns false!

Definition at line 171 of file aword_index.hpp.

8.2.3.9 bool uva::smt::bpbd::server::lm::dictionary::aword_index::is_word_registering_needed ( ) const [inline]

This method allows to indicate whether registering a word is needed by the given implementation of the word index.

Returns

true if the word registering is needed, otherwise false.

Definition at line 96 of file aword_index.hpp.

8.2.3.10 word_uid uva::smt::bpbd::server::lm::dictionary::aword_index::register_word (const text_piece_reader & *token*) [inline]

This function creates/gets an id for the given word. Note: The ids must be unique and continuous! The returned word id is always >= MIN_KNOWN_WORD_ID

Parameters

token	the word to hash

Returns

the resulting hash

Definition at line 107 of file aword_index.hpp.

8.2.3.11 void uva::smt::bpbd::server::lm::dictionary::aword_index::reserve (const size_t *num_words*) [inline]

This method should be used to pre-allocate the word index

Parameters

num_words	the number of words

Definition at line 67 of file aword_index.hpp.

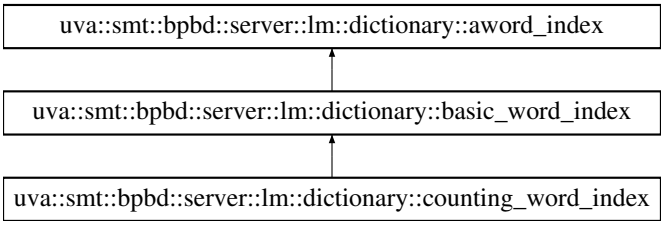
The documentation for this class was generated from the following file:

- inc/server/lm/dictionaries/aword_index.hpp

8.3 uva::smt::bpbd::server::lm::dictionary::basic_word_index Class Reference

`#include <basic_word_index.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::dictionary::basic_word_index:



Public Types

- typedef pair< const string, word_uid > TWordIndexEntry
- typedef greedy_memory_allocator< TWordIndexEntry > TWordIndexAllocator
- typedef unordered_map< string, word_uid, std::hash< string >, std::equal_to< string >, TWordIndex↩ Allocator > TWordIndexMap
- typedef TWordIndexMap::const_iterator TWordIndexMapConstIter

Public Member Functions

- basic_word_index (const float wordIndexMemFactor)
- size_t get_number_of_words (const size_t num_words) const
- void reserve (const size_t num_words)
- word_uid get_word_id (const text_piece_reader &token) const
- bool is_word_registering_needed () const
- word_uid register_word (const text_piece_reader &token)
- bool is_word_counts_needed () const
- void count_word (const text_piece_reader &word, prob_weight prob)
- void do_post_word_count ()
- bool is_post_actions_needed () const
- void do_post_actions ()
- virtual ~basic_word_index ()
- TWordIndexMapConstIter begin ()
- TWordIndexMapConstIter end ()

Static Public Member Functions

- static constexpr bool is_word_index_continuous ()

Protected Member Functions

- basic_word_index (const basic_word_index &other)

Protected Attributes

- TWordIndexAllocator * m_word_index_alloc_ptr
- TWordIndexMap * m_word_index_map_ptr
- word_uid m_next_new_word_id
- const float m_word_index_mem_factor

8.3.1 Detailed Description

This is a hash-map based implementation of the word index.

Definition at line 57 of file basic_word_index.hpp.

8.3.2 Member Typedef Documentation

8.3.2.1 typedef greedy_memory_allocator< TWordIndexEntry > uva::smt::bpbd::server::lm::dictionary::basic_word_index::TWordIndexAllocator

The typedef for the word index allocator

Definition at line 212 of file basic_word_index.hpp.

8.3.2.2 typedef pair< const string, word_uid> uva::smt::bpbd::server::lm::dictionary::basic_word_index::TWordIndexEntry

The type of key,value pairs to be stored in the word index

Definition at line 202 of file basic_word_index.hpp.

8.3.2.3 typedef unordered_map<string, word_uid, std::hash<string>, std::equal_to<string>, TWordIndexAllocator > uva::smt::bpbd::server::lm::dictionary::basic_word_index::TWordIndexMap

The word index map type

Definition at line 217 of file basic_word_index.hpp.

8.3.2.4 typedef TWordIndexMap::const_iterator uva::smt::bpbd::server::lm::dictionary::basic_word_index::TWordIndexMapConstIter

Defines the constant iterator type

Definition at line 222 of file basic_word_index.hpp.

8.3.3 Constructor & Destructor Documentation

8.3.3.1 uva::smt::bpbd::server::lm::dictionary::basic_word_index::basic_word_index (const float *wordIndexMemFactor*) [inline]

The basic constructor

Parameters

wordIndexMemFactor	the assigned memory factor for storage allocation in the unordered_map used for the word index

Definition at line 65 of file basic_word_index.hpp.

8.3.3.2 virtual uva::smt::bpbd::server::lm::dictionary::basic_word_index::~basic_word_index ( ) [inline], [virtual]

The basic destructor

Definition at line 200 of file basic_word_index.hpp.

8.3.3.3 uva::smt::bpbd::server::lm::dictionary::basic_word_index::basic_word_index (const basic_word_index & *other*) [inline], [protected]

The copy constructor, is made private as we do not intend to copy this class objects

Parameters

orig	the object to copy from

Definition at line 246 of file basic_word_index.hpp.

8.3.4 Member Function Documentation

8.3.4.1 TWordIndexMapConstIter uva::smt::bpbd::server::lm::dictionary::basic_word_index::begin ( ) [inline]

Allows to get the begin constant iterator

Returns

the begin constant iterator

Definition at line 228 of file basic_word_index.hpp.

8.3.4.2 void uva::smt::bpbd::server::lm::dictionary::basic_word_index::count_word (const text_piece_reader & *word*, prob_weight *prob*) [inline]

This method is to be used when the word counting is needed.

See also

AWordIndex

Definition at line 152 of file basic_word_index.hpp.

8.3.4.3 void uva::smt::bpbd::server::lm::dictionary::basic_word_index::do_post_actions ( ) [inline]

Is to be called if the post actions are needed right after that all the individual words have been added into the index.

See also

AWordIndex

Definition at line 182 of file basic_word_index.hpp.

8.3.4.4 void uva::smt::bpbd::server::lm::dictionary::basic_word_index::do_post_word_count ( ) [inline]

Should be called if the word count is needed after all the words have been counted.

See also

AWordIndex

Definition at line 162 of file basic_word_index.hpp.

8.3.4.5 TWordIndexMapConstIter uva::smt::bpbd::server::lm::dictionary::basic_word_index::end ( ) [inline]

Allows to get the end constant iterator

Returns

the end constant iterator

Definition at line 236 of file basic_word_index.hpp.

8.3.4.6 size_t uva::smt::bpbd::server::lm::dictionary::basic_word_index::get_number_of_words (const size_t *num_words*) const [inline]

Allows to get the total words count including the unknown and undefined words

See also

AWordIndex

Definition at line 73 of file basic_word_index.hpp.

8.3.4.7 word_uid uva::smt::bpbd::server::lm::dictionary::basic_word_index::get_word_id (const text_piece_reader & *token*) const [inline]

This function gets an id for the given word word based no the stored 1-Grams. If the word is not known then an unknown word ID is returned: UNKNOWN_WORD_ID

See also

AWordIndex

Definition at line 100 of file basic_word_index.hpp.

8.3.4.8 bool uva::smt::bpbd::server::lm::dictionary::basic_word_index::is_post_actions_needed ( ) const [inline]

Indicates if the post-actions are needed. The post actions should be called after all the words have been filled into the index.

See also

AWordIndex

Definition at line 173 of file basic_word_index.hpp.

8.3.4.9 bool uva::smt::bpbd::server::lm::dictionary::basic_word_index::is_word_counts_needed ( ) const [inline]

This method allows to indicate whether word counting is needed by the given implementation of the word index.

See also

AWordIndex

Definition at line 144 of file basic_word_index.hpp.

8.3.4.10 static constexpr bool uva::smt::bpbd::server::lm::dictionary::basic_word_index::is_word_index_continuous ( ) [inline], [static]

Allows to indicate if the word index is continuous, i.e. it issues the word ids in a continuous range starting from 0.

See also

AWordIndex

Returns

true - this word index is continuous.

Definition at line 193 of file basic_word_index.hpp.

8.3.4.11 bool uva::smt::bpbd::server::lm::dictionary::basic_word_index::is_word_registering_needed ( ) const [inline]

This method allows to indicate whether registering a word is needed by the given implementation of the word index.

See also

AWordIndex

Definition at line 117 of file basic_word_index.hpp.

8.3.4.12 word_uid uva::smt::bpbd::server::lm::dictionary::basic_word_index::register_word (const text_piece_reader & *token*) [inline]

This function creates/gets a hash for the given word.

See also

AWordIndex

Definition at line 125 of file basic_word_index.hpp.

8.3.4.13 void uva::smt::bpbd::server::lm::dictionary::basic_word_index::reserve (const size_t *num_words*) [inline]

This method should be used to pre-allocate the word index

See also

AWordIndex

Definition at line 81 of file basic_word_index.hpp.

8.3.5 Member Data Documentation

8.3.5.1 word_uid uva::smt::bpbd::server::lm::dictionary::basic_word_index::m_next_new_word_id [protected]

Definition at line 258 of file basic_word_index.hpp.

8.3.5.2 TWordIndexAllocator* uva::smt::bpbd::server::lm::dictionary::basic_word_index::m_word_index_alloc_ptr [protected]

Definition at line 252 of file basic_word_index.hpp.

8.3.5.3 TWordIndexMap* uva::smt::bpbd::server::lm::dictionary::basic_word_index::m_word_index_map_ptr [protected]

Definition at line 255 of file basic_word_index.hpp.

8.3.5.4 const float uva::smt::bpbd::server::lm::dictionary::basic_word_index::m_word_index_mem_factor [protected]

Definition at line 262 of file basic_word_index.hpp.

The documentation for this class was generated from the following file:

- inc/server/lm/dictionaries/basic_word_index.hpp

8.4 uva::smt::bpbd::server::lm::caching::BitmapHashCache Class Reference

`#include <bitmap_hash_cache.hpp>`

Public Member Functions

- BitmapHashCache ()
- virtual ~BitmapHashCache ()
- void pre_allocate (const size_t num_elems, const uint8_t buckets_factor)
- void cache_m_gram_hash (const model_m_gram gram)
- bool is_hash_cached (uint_fast64_t key) const

8.4.1 Detailed Description

This class is to be used for caching the present of M-grams in the trie. The way it is done is using a bitset. A bitset indicates which hashes are present and therefore which M-grams have a chance to be found in the Trie. This class can give potential speed improvement for the Tries which are context/layer based and use search algorithms to go through levels.

Definition at line 61 of file bitmap_hash_cache.hpp.

8.4.2 Constructor & Destructor Documentation

8.4.2.1 uva::smt::bpbd::server::lm::caching::BitmapHashCache::BitmapHashCache ( ) [inline]

The basic constructor, does not do much - only default initialization

Definition at line 67 of file bitmap_hash_cache.hpp.

8.4.2.2 virtual uva::smt::bpbd::server::lm::caching::BitmapHashCache::~BitmapHashCache ( ) [inline], [virtual]

The basic destructor

Definition at line 73 of file bitmap_hash_cache.hpp.

8.4.3 Member Function Documentation

8.4.3.1 void uva::smt::bpbd::server::lm::caching::BitmapHashCache::cache_m_gram_hash (const model_m_gram *gram*) [inline]

Allows to add the M-gram to the cache

Parameters

gram	the M-gram to cache

Definition at line 110 of file bitmap_hash_cache.hpp.

8.4.3.2 bool uva::smt::bpbd::server::lm::caching::BitmapHashCache::is_hash_cached (uint_fast64_t *key*) const [inline]

Allows to check if the given sub-m-gram, defined by the begin_word_idx and end_word_idx parameters, is potentially present in the trie.

Parameters

key	the m-gram key

Returns

true if the sub-m-gram is potentially present, otherwise false

Definition at line 131 of file bitmap_hash_cache.hpp.

8.4.3.3 void uva::smt::bpbd::server::lm::caching::BitmapHashCache::pre_allocate (const size_t *num_elems*, const uint8_t *buckets_factor*) [inline]

Allowo to pre-allocate memory for the bitset

Parameters

num_elems	

Returns

Definition at line 84 of file bitmap_hash_cache.hpp.

The documentation for this class was generated from the following file:

- inc/server/lm/models/bitmap_hash_cache.hpp

8.5 uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType > Class Template Reference

`#include <m_gram_id.hpp>`

Static Public Member Functions

- static uint8_t create_m_gram_id (const TWordIdType *word_ids, const uint8_t num_word_ids, TM_Gram_Id_Value_Ptr &m_p_gram_id)
- static uint8_t compute_m_gram_id (const TWordIdType *word_ids, const uint8_t num_word_ids, TM_Gram_Id_Value_Ptr m_p_gram_id)
- static void allocate_byte_m_gram_id (const phrase_length level, TM_Gram_Id_Value_Ptr &m_p_gram_id)
- static int compare (const uint8_t id_len_bytes, const TM_Gram_Id_Value_Ptr &m_p_gram_id_one, const TM_Gram_Id_Value_Ptr &m_p_gram_id_two)
- static bool is_equal_m_grams_id (const uint8_t id_len_bytes, const TM_Gram_Id_Value_Ptr &one, const TM_Gram_Id_Value_Ptr &two)
- static bool is_less_m_grams_id (const uint8_t id_len_bytes, const TM_Gram_Id_Value_Ptr &one, const TM_Gram_Id_Value_Ptr &two)
- template<phrase_length CURR_LEVEL>
  static const uint8_t & gram_id_type_2_byte_len (uint32_t id_type)
- static const uint32_t & gram_id_byte_len_2_type (const phrase_length gram_level, uint8_t *len_bytes)
- template<phrase_length CURR_LEVEL>
  static bool is_less_m_grams_id (const uint8_t id_type_len_bytes, const TM_Gram_Id_Value_Ptr &one, const TM_Gram_Id_Value_Ptr &two)
- static bool is_more_m_grams_id (const uint8_t id_len_bytes, const TM_Gram_Id_Value_Ptr &one, const TM_Gram_Id_Value_Ptr &two)

Static Public Attributes

- static constexpr uint8_t NUM_BYTES_WORD_ID = sizeof (TWordIdType)
- static constexpr uint32_t NUMBER_ID_TYPES_PER_LEVEL [ ]
- static constexpr uint8_t ID_TYPE_LEN_BYTES [ ]
- static constexpr uint8_t MAX_ID_LEN_BYTES [ ]
- static constexpr uint32_t LEVEL_2_GRAM_TO_TYPE_LEN [NUM_BYTES_WORD_ID][NUM_BYTES_W↩ ORD_ID]
- static constexpr uint32_t LEVEL_3_GRAM_TO_TYPE_LEN [NUM_BYTES_WORD_ID][NUM_BYTES_W↩ ORD_ID][NUM_BYTES_WORD_ID]
- static constexpr uint32_t LEVEL_4_GRAM_TO_TYPE_LEN [NUM_BYTES_WORD_ID][NUM_BYTES_W↩ ORD_ID][NUM_BYTES_WORD_ID][NUM_BYTES_WORD_ID]
- static constexpr uint32_t LEVEL_5_GRAM_TO_TYPE_LEN [NUM_BYTES_WORD_ID][NUM_BYTES_W↩ ORD_ID][NUM_BYTES_WORD_ID][NUM_BYTES_WORD_ID][NUM_BYTES_WORD_ID]
- static constexpr uint32_t LEVEL_6_GRAM_TO_TYPE_LEN [NUM_BYTES_WORD_ID][NUM_BYTES_W↩ ORD_ID][NUM_BYTES_WORD_ID][NUM_BYTES_WORD_ID][NUM_BYTES_WORD_ID][NUM_BYTES↩ _WORD_ID]
- static constexpr uint8_t LEVEL_2_GRAM_TO_BYTE_LEN [ ] = {2, 3, 4, 5, 3, 4, 5, 6, 4, 5, 6, 7, 5, 6, 7, 8}
- static constexpr uint8_t LEVEL_3_GRAM_TO_BYTE_LEN [ ] = {3, 4, 5, 6, 4, 5, 6, 7, 5, 6, 7, 8, 6, 7, 8, 9, 4, 5, 6, 7, 5, 6, 7, 8, 6, 7, 8, 9, 7, 8, 9, 10, 5, 6, 7, 8, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12}
- static constexpr uint8_t LEVEL_4_GRAM_TO_BYTE_LEN [ ] = {4, 5, 6, 7, 5, 6, 7, 8, 6, 7, 8, 9, 7, 8, 9, 10, 5, 6, 7, 8, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 5, 6, 7, 8, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16}
- static constexpr uint8_t LEVEL_5_GRAM_TO_BYTE_LEN [ ] = {5, 6, 7, 8, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15,

13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 17, 18, 19, 20}

- static constexpr uint8_t LEVEL_6_GRAM_TO_BYTE_LEN [ ] = {6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14,

12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 17, 18, 19, 20, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 17, 18, 19, 20, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 17, 18, 19, 20, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 17, 18, 19, 20, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 17, 18, 19, 20, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19

15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 17, 18, 19, 20, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 17, 18, 19, 20, 15, 16, 17, 18, 16, 17, 18, 19, 17, 18, 19, 20, 18, 19, 20, 21, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 17, 18, 19, 20, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 17, 18, 19, 20, 15, 16, 17, 18, 16, 17, 18, 19, 17, 18, 19, 20, 18, 19, 20, 21, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 17, 18, 19, 20, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 17, 18, 19, 20, 15, 16, 17, 18, 16, 17, 18, 19, 17, 18, 19, 20, 18, 19, 20, 21}

8.5.1 Detailed Description

template<typename TWordIdType>class uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >

The byte-compressed implementation of the M-gram id class

Definition at line 97 of file m_gram_id.hpp.

8.5.2 Member Function Documentation

8.5.2.1 template<typename TWordIdType > static void uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::allocate_byte_m_gram_id (const phrase_length *level,* TM_Gram_Id_Value_Ptr & *m_p_gram_id*) [inline], [static]

The basic constructor that allocates maximum memory needed to store the M-gram id of the given level.

Parameters

level	the level of the M-grams this object will store id for.
m_p_gram_id	the pointer to initialize

Definition at line 192 of file m_gram_id.hpp.

8.5.2.2 template<typename TWordIdType > static int uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_↩ M_Gram_Id< TWordIdType >::compare (const uint8_t *id_len_bytes,* const TM_Gram_Id_Value_Ptr & *m_p_gram_id_one,* const TM_Gram_Id_Value_Ptr & *m_p_gram_id_two*) [inline],[static]

Allows to compare two M-Gram ids of a fixed M-gram level

Parameters

id_len_bytes	the minimum total number of bytes in both m-gram ids.
m_p_gram_id_one	the first M-gram id
m_p_gram_id_two	the second M-gram id

Returns

Negative value if one is smaller than two Zero if one is equal to two Positive value if one is larger than two

Definition at line 213 of file m_gram_id.hpp.

8.5.2.3 template<typename TWordIdType > uint8_t uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::compute_m_gram_id (const TWordIdType * *word_ids*, const uint8_t *num_word_ids*, TM_Gram_Id_Value_Ptr *m_p_gram_id*) [static]

This method allows to compute the m-gram id and set it into the pre-allocated memory given by a pointer.

Parameters

word_ids	the pointer to the array of word ids
num_word_ids	the number of word ids
m_p_gram_id	the pointer to the data storage to be initialized

Returns

the number of bytes in the m-gram id

This method is needed to compute the M-gram id.

Let us give an example of a 2-gram id for a given 2-gram:

1) The 2 word_ids are to be converted to the 2-gram id: There are 4 bytes in one word id and 4 bytes in another word id, In total we have 4^2 possible 2-gram id lengths in bytes, if we only use meaningful bytes of the word id for instance: 01-01 both really need just two bytes 01-02 the first needs one and another two 02-01 the first needs two and another one ... 04-04 both need 8 bytes

2) These 4^2 = 16 combinations uniquely identify the type of stored id. So this can be an uid of the gram id type. To store such a uid we need ceil(log2(16)/8)= 1 bits.

3) We create the 2-gram id as a byte array of 1+ bytes: the type

- the meaningful byte from wordId2 and wordId1. We start from the end (reverse the word order) as this can potentially increase speed of the comparison operation.

Definition at line 252 of file byte_m_gram_id.cpp.

8.5.2.4 template<typename TWordIdType > uint8_t uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::create_m_gram_id (const TWordIdType * *word_ids*, const uint8_t *num_word_ids*, TM_Gram_Id_Value_Ptr & *m_p_gram_id*) [static]

This method allows to create new M-gram id for the given M-gram. There should be no memory allocated for the M-gram id. This method will allocate as much as needed to store the given id.

Parameters

word_ids	the pointer to the array of word ids
num_word_ids	the number of word ids
m_p_gram_id	the pointer to the data storage to be initialized

Returns

the number of bytes in the m-gram id

This method is needed to compute the M-gram id.

Let us give an example of a 2-gram id for a given 2-gram:

1) The 2 word_ids are to be converted to the 2-gram id: There are 4 bytes in one word id and 4 bytes in another word id, In total we have 4^2 possible 2-gram id lengths in bytes, if we only use meaningful bytes of the word id for instance: 01-01 both really need just two bytes 01-02 the first needs one and another two 02-01 the first needs two and another one ... 04-04 both need 8 bytes

2) These 4^2 = 16 combinations uniquely identify the type of stored id. So this can be an uid of the gram id type. To store such a uid we need ceil(log2(16)/8)= 1 bits.

3) We create the 2-gram id as a byte array of 1+ bytes: the type

- the meaningful byte from wordId2 and wordId1. We start from the end (reverse the word order) as this can potentially increase speed of the comparison operation.

Definition at line 173 of file byte_m_gram_id.cpp.

8.5.2.5 template<typename TWordIdType > static const uint32_t& uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::gram_id_byte_len_2_type (const phrase_length *gram_level,* uint8_t * *len_bytes*) [inline], [static]

This method is needed to compute the id type identifier. Can compute the id type for the M-grams until (and including) M = 6

Parameters

gram_level	the number of word ids
len_bytes	the bytes needed per word id
return	the resulting id type the initial value is expected to be 0

Definition at line 288 of file m_gram_id.hpp.

8.5.2.6 template<typename TWordIdType > template<phrase_length CURR_LEVEL> static const uint8_t& uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::gram_id_type_2_byte_len (uint32_t *id_type*) [inline], [static]

Allows to compute the byte length for the id of the given type. Can compute the byte length for the M-grams until (and including) M = 6.

Parameters

CURR_LEVEL	the M-Gram level M
id_type	the type id

Returns

the total byte length to store the id of this type.

Definition at line 259 of file m_gram_id.hpp.

8.5.2.7 template<typename TWordIdType > static bool uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::is_equal_m_grams_id (const uint8_t *id_len_bytes,* const TM_Gram_Id_Value_Ptr & *one,* const TM_Gram_Id_Value_Ptr & *two*) [inline], [static]

This is a fore-declaration of the function that can compare two M-gram ids of the same given level

Parameters

id_len_bytes	the minimum total number of bytes in both m-gram ids.
one	the first M-gram to compare
two	the second M-gram to compare

Returns

true if the first M-gram is "smaller" than the second, otherwise false

Definition at line 234 of file m_gram_id.hpp.

8.5.2.8 template<typename TWordIdType > static bool uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::is_less_m_grams_id (const uint8_t *id_len_bytes,* const TM_Gram_Id_Value_Ptr & *one,* const TM_Gram_Id_Value_Ptr & *two*) [inline],[static]

This is a fore-declaration of the function that can compare two M-gram ids of the same given level

Parameters

id_len_bytes	the minimum total number of bytes in both m-gram ids.
one	the first M-gram to compare
two	the second M-gram to compare

Returns

true if the first M-gram is "smaller" than the second, otherwise false

Definition at line 246 of file m_gram_id.hpp.

8.5.2.9 template<typename TWordIdType > template<phrase_length CURR_LEVEL> static bool uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::is_less_m_grams_id (const uint8_t *id_type_len_bytes,* const TM_Gram_Id_Value_Ptr & *one,* const TM_Gram_Id_Value_Ptr & *two*) [inline],[static]

This is a fore-declaration of the function that can compare two M-gram ids of the same given level

Parameters

id_type_len_bytes	the minimum total number of bytes in both m-gram ids.
one	the first M-gram to compare
two	the second M-gram to compare

Returns

true if the first M-gram is "smaller" than the second, otherwise false

Definition at line 318 of file m_gram_id.hpp.

8.5.2.10 template<typename TWordIdType > static bool uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::is_more_m_grams_id (const uint8_t *id_len_bytes,* const TM_Gram_Id_Value_Ptr & *one,* const TM_Gram_Id_Value_Ptr & *two*) [inline],[static]

This is a fore-declaration of the function that can compare two M-gram ids of the same given level

Parameters

id_len_bytes	the minimum total number of bytes in both m-gram ids.
one	the first M-gram to compare
two	the second M-gram to compare

Returns

true if the first M-gram is "larger" than the second, otherwise false

Definition at line 347 of file m_gram_id.hpp.

8.5.3 Member Data Documentation

8.5.3.1 template<typename TWordIdType > constexpr uint8_t uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::ID_TYPE_LEN_BYTES [static]

Initial value:

```
= {
                                    0,
                                    N_GRAM_ID_TYPE_LEN_BYTES(1),
                                    N_GRAM_ID_TYPE_LEN_BYTES(2),
                                    N_GRAM_ID_TYPE_LEN_BYTES(3),
                                    N_GRAM_ID_TYPE_LEN_BYTES(4),
                                    N_GRAM_ID_TYPE_LEN_BYTES(5),
                                    N_GRAM_ID_TYPE_LEN_BYTES(6),

                                }
```

Definition at line 125 of file m_gram_id.hpp.

8.5.3.2 template<typename TWordIdType > constexpr uint8_t uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::LEVEL_2_GRAM_TO_BYTE_LEN = {2, 3, 4, 5, 3, 4, 5, 6, 4, 5, 6, 7, 5, 6, 7, 8} [static]

Definition at line 2362 of file m_gram_id.hpp.

8.5.3.3 template<typename TWordIdType > constexpr uint32_t uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::LEVEL_2_GRAM_TO_TYPE_LEN [static]

Initial value:

```
= {
    { 0, 4, 8, 12},
    { 1, 5, 9, 13},
    { 2, 6, 10, 14},
    { 3, 7, 11, 15}
}
```

This method is needed to compute the id type identifier. Can compute the id type for M-grams until (including) M = 5 The type is computed as in a 32-based numeric system, e.g. for M==5: (len_bits[0]-1)∗32^0 + (len_bits[1]-1)∗32^1 + (len_bits[2]-1)∗32^2 + (len_bits[3]-1)∗32^3 + (len_bits[4]-1)∗32^4

Parameters

gram_level	the number of word ids
len_bytes	the bytes needed per word id
id_type	[out] the resulting id type the initial value is expected to be 0

static inline void gram_id_byte_len_2_type(const phrase_length gram_level, uint8_t * len_bytes, uint32_t & id_↩ type) { Do the sanity check for against overflows ASSERT_SANITY_THROW((gram_level > M_GRAM_LEVEL_5), string("Unsupported m-gram level: ") + std::to_string(gram_level) + string(", must be within [") + std::to_string(M_↩ GRAM_LEVEL_2) + string(", ") + std::to_string(M_GRAM_LEVEL_6) + string("], insufficient multipliers!"));

LOG_DEBUG3 << "Computing the " << SSTR(gram_level) << "-gram id type" << END_LOG;

Compute the M-gram id type. Here we use the pre-computed multipliers for (size_t idx = 0; idx < gram_level; ++idx) { LOG_DEBUG3 << ((uint32_t) len_bytes[idx] - 1) << " * " << NUMBER_ID_TYPES_PER_LEVEL[idx] << " = " << ((uint32_t) len_bytes[idx] - 1) * NUMBER_ID_TYPES_PER_LEVEL[idx] << END_LOG;

id_type += ((uint32_t) len_bytes[idx] - 1) * NUMBER_ID_TYPES_PER_LEVEL[idx]; } LOG_DEBUG3 << "↩ Resulting id_type = " << SSTR(id_type) << END_LOG; };

Definition at line 61 of file m_gram_id.hpp.

8.5.3.4 template<typename TWordIdType > constexpr uint8_t uva::smt::bpbd::server::lm::m_grams::m_gram_id↩ ::Byte_M_Gram_Id< TWordIdType >::LEVEL_3_GRAM_TO_BYTE_LEN = {3, 4, 5, 6, 4, 5, 6, 7, 5, 6, 7, 8, 6, 7, 8, 9, 4, 5, 6, 7, 5, 6, 7, 8, 6, 7, 8, 9, 7, 8, 9, 10, 5, 6, 7, 8, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12} [static]

Definition at line 2363 of file m_gram_id.hpp.

8.5.3.5 template<typename TWordIdType > constexpr uint32_t uva::smt::bpbd::server::lm::m_↩ grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::LEVEL_3_GRAM_TO_TYPE_LEN [static]

Initial value:

```
= {
    {
        { 0, 16, 32, 48},
        { 4, 20, 36, 52},
        { 8, 24, 40, 56},
        { 12, 28, 44, 60}
    },
    {
        { 1, 17, 33, 49},
        { 5, 21, 37, 53},
        { 9, 25, 41, 57},
        { 13, 29, 45, 61}
    },
    {
        { 2, 18, 34, 50},
        { 6, 22, 38, 54},
        { 10, 26, 42, 58},
        { 14, 30, 46, 62}
    },
    {
        { 3, 19, 35, 51},
        { 7, 23, 39, 55},
        { 11, 27, 43, 59},
        { 15, 31, 47, 63}
    }
}
```

Definition at line 67 of file m_gram_id.hpp.

8.5.3.6 template<typename TWordIdType > constexpr uint8_t uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::LEVEL_4_GRAM_TO_BYTE_LEN = {4, 5, 6, 7, 5, 6, 7, 8, 6, 7, 8, 9, 7, 8, 9, 10, 5, 6, 7, 8, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 5, 6, 7, 8, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, [static]

Definition at line 2364 of file m_gram_id.hpp.

8.5.3.7 template<typename TWordIdType > constexpr uint32_t uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::LEVEL_4_GRAM_TO_TYPE_LEN [static]

Definition at line 93 of file m_gram_id.hpp.

8.5.3.8 template<typename TWordIdType > constexpr uint8_t uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::LEVEL_5_GRAM_TO_BYTE_LEN = {5, 6, 7, 8, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 14, 15, 16, 17, 15, 16, 17, 18, 16, 17, 18, 19, 17, 18, 19, 20} [static]

Definition at line 2365 of file m_gram_id.hpp.

8.5.3.9 template<typename TWordIdType > constexpr uint32_t uva::smt::bpbd::server::lm::m_↩ grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::LEVEL_5_GRAM_TO_TYPE_LEN [static]

Definition at line 199 of file m_gram_id.hpp.

8.5.3.10 template<typename TWordIdType > constexpr uint8_t uva::smt::bpbd::server::lm::m_grams::m_gram_↩ id::Byte_M_Gram_Id< TWordIdType >::LEVEL_6_GRAM_TO_BYTE_LEN = {6, 7, 8, 9, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 7, 8, 9, 10, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 8, 9, 10, 11, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 9, 10, 11, 12, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 10, 11, 12, 13, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 11, 12, 13, 14, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 12, 13, 14, 15, 13, 14, 15, 16, 14, 15, 16, 17, 15, 16, 17, 18,

8.5.3.11 template<typename TWordIdType > constexpr uint32_t uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::LEVEL_6_GRAM_TO_TYPE_LEN [static]

Definition at line 625 of file m_gram_id.hpp.

8.5.3.12 template<typename TWordIdType > constexpr uint8_t uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::MAX_ID_LEN_BYTES [static]

Initial value:

```
= {
                                    0,
                                    MAX_N_GRAM_ID_LEN_BYTES(1),
                                    MAX_N_GRAM_ID_LEN_BYTES(2),
                                    MAX_N_GRAM_ID_LEN_BYTES(3),
                                    MAX_N_GRAM_ID_LEN_BYTES(4),
                                    MAX_N_GRAM_ID_LEN_BYTES(5),
                                    MAX_N_GRAM_ID_LEN_BYTES(6),

                                }
```

Definition at line 146 of file m_gram_id.hpp.

8.5.3.13 template<typename TWordIdType > constexpr uint8_t uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::NUM_BYTES_WORD_ID = sizeof (TWordIdType) [static]

Definition at line 101 of file m_gram_id.hpp.

8.5.3.14 template<typename TWordIdType > constexpr uint32_t uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >::NUMBER_ID_TYPES_PER_LEVEL [static]

Initial value:

```
= {
                                    const_expr::power(
      NUM_BYTES_WORD_ID, 0),
                                    const_expr::power(
      NUM_BYTES_WORD_ID, 1),
                                    const_expr::power(
      NUM_BYTES_WORD_ID, 2),
                                    const_expr::power(
      NUM_BYTES_WORD_ID, 3),
                                    const_expr::power(
      NUM_BYTES_WORD_ID, 4),
                                    const_expr::power(
      NUM_BYTES_WORD_ID, 5),
                                    const_expr::power(
      NUM_BYTES_WORD_ID, 6),
                                    const_expr::power(
      NUM_BYTES_WORD_ID, 7)
                                }
```

Stores the m-gram id multipliers multipliers up to and including level 7

Definition at line 106 of file m_gram_id.hpp.

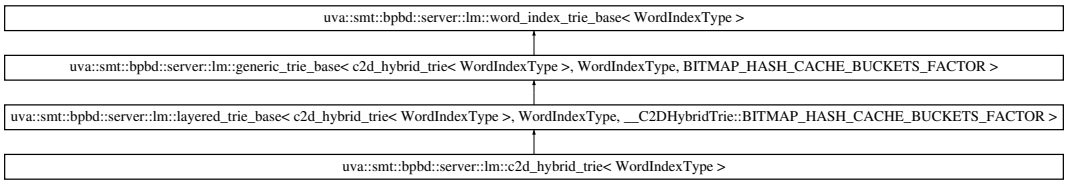
The documentation for this class was generated from the following files:

- inc/server/lm/mgrams/m_gram_id.hpp
- src/server/lm/mgrams/byte_m_gram_id.cpp

8.6 uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType > Class Template Reference

`#include <c2d_hybrid_trie.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType >:



Public Types

- typedef layered_trie_base< c2d_hybrid_trie< WordIndexType >, WordIndexType, __C2DHybridTrie::BIT↩ MAP_HASH_CACHE_BUCKETS_FACTOR > BASE

Public Member Functions

- c2d_hybrid_trie (WordIndexType &word_index, const float mram_mem_factor=__C2DHybridTrie::UM_M_↩ GRAM_MEMORY_FACTOR, const float ngram_mem_factor=__C2DHybridTrie::UM_N_GRAM_MEMORY↩ _FACTOR)
- float get_unk_word_prob () const
- bool get_ctx_id (const phrase_length level_idx, const TShortId word_id, TLongId &ctx_id) const
- void log_model_type_info () const
- virtual void pre_allocate (const size_t counts[LM_M_GRAM_LEVEL_MAX])
- template<phrase_length CURR_LEVEL>
  void add_m_gram (const model_m_gram &gram)
- void get_unigram_payload (m_gram_query &query) const
- void get_m_gram_payload (m_gram_query &query, MGramStatusEnum &status) const
- void get_n_gram_payload (m_gram_query &query, MGramStatusEnum &status) const
- void set_def_unk_word_prob (const prob_weight prob)
- virtual ~c2d_hybrid_trie ()

Additional Inherited Members

8.6.1 Detailed Description

template<typename WordIndexType>class uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType >

This is a hybrid trie implementation inspired by the four other ones:

W2COrderedArrayTrie, C2WOrderedArrayTrie, CtxMultiHashMapTrie, and W2CHybridMemoryTrie

It tries to be as much memory efficient as speed efficient. More specifically we store as much data as possible in an array form in order to get optimal memory consumption and having short and easily definable context index. Yet, we use unordered maps for the sake of speeding up queries, as they allow us to realize (word_id, ctx_id) to ctx_id in the most efficient manner. The lookup should be just O(1) whereas in the lookup is O(log(n)), as we need to use binary searches there.

Definition at line 68 of file c2d_hybrid_trie.hpp.

8.6.2 Member Typedef Documentation

8.6.2.1 template<typename WordIndexType > typedef layered_trie_base<c2d_hybrid_trie<WordIndexType>, WordIndexType, __C2DHybridTrie::BITMAP_HASH_CACHE_BUCKETS_FACTOR> uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType >::BASE

Definition at line 70 of file c2d_hybrid_trie.hpp.

8.6.3 Constructor & Destructor Documentation

8.6.3.1 template<typename WordIndexType > uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType >::c2d_hybrid_trie (WordIndexType & *word_index,* const float *mram_mem_factor* = __C2DHybridTrie::UM_M_GRAM_MEMORY_FACTOR, const float *ngram_mem_factor* = __C2DHybridTrie::UM_N_GRAM_MEMORY_FACTOR) [explicit]

The basic class constructor, accepts memory factors that are the coefficients used when pre-allocating memory for unordered maps.

If a factor is equal to 0.0 then no memory is pre-allocated. If the factor is equal to 1.0 then there is only as much preallocated as needed to store the gram entries. The latter is typically not enough as unordered_map needs more memory for internal administration. If there is not enough memory pre-allocated then additional allocations will take place but it does not alway lead to more efficient memory usage. The observed behavior is that it is better to pre-allocate a bit more memory beforehand, than needed. This leads to less memory consumption. Depending on the type of unordered_map key/value pair types the advised factor values are from 2.0 to 2.6. Because it can not be optimally determined beforehand, these are made constructor parameters so that they can be configured by the used. This breaks encapsulation a bit, exposing the internals, but there is no other better way, for fine tuning the memory usage.

Parameters

_pWordIndex	the word index to be used
_oGramMem↩ Factor	The One-Gram memory factor needed for the greedy allocator for the unordered_map
_mGramMem↩ Factor	The M-Gram memory factor needed for the greedy allocator for the unordered_map
_nGramMem↩ Factor	The N-Gram memory factor needed for the greedy allocator for the unordered_map

Definition at line 48 of file c2d_hybrid_trie.cpp.

8.6.3.2 template<typename WordIndexType > uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType >::~c2d_hybrid_trie ( ) [virtual]

The basic destructor

Definition at line 146 of file c2d_hybrid_trie.cpp.

8.6.4 Member Function Documentation

8.6.4.1 template<typename WordIndexType > template<phrase_length CURR_LEVEL> void uva::smt::bpbd↩ ::server::lm::c2d_hybrid_trie< WordIndexType >::add_m_gram (const model_m_gram & *gram*) [inline]

Allows to retrieve the data storage structure for the M gram with the given M-gram level Id. M-gram context and last word Id. If the storage structure does not exist, return a new one. For more details

See also

LayeredTrieBase

Definition at line 152 of file c2d_hybrid_trie.hpp.

8.6.4.2 template<typename WordIndexType > bool uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType >::get_ctx_id (const phrase_length *level_idx,* const TShortId *word_id,* TLongId & *ctx_id*) const [inline]

Computes the M-Gram context using the previous context and the current word id

See also

LayeredTrieBese

Definition at line 115 of file c2d_hybrid_trie.hpp.

8.6.4.3 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType >::get_m_gram_payload (m_gram_query & *query,* MGramStatusEnum & *status*) const [inline]

Allows to retrieve the payload for the M-gram defined by the end word_id and ctx_id.

See also

GenericTrieBase

Definition at line 206 of file c2d_hybrid_trie.hpp.

8.6.4.4 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType >::get_n_gram_payload (m_gram_query & *query,* MGramStatusEnum & *status*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for m==n

See also

GenericTrieBase

Definition at line 244 of file c2d_hybrid_trie.hpp.

8.6.4.5 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType >::get_unigram_payload (m_gram_query & *query*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for m==1. The retrieval of a uni-gram data is always a success

See also

GenericTrieBase

Definition at line 191 of file c2d_hybrid_trie.hpp.

8.6.4.6 template<typename WordIndexType > float uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType >::get_unk_word_prob ( ) const [inline]

Allows to retrieve the unknown target word log probability penalty

Returns

the target source word log probability penalty

Definition at line 107 of file c2d_hybrid_trie.hpp.

8.6.4.7 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType >::log_model_type_info ( ) const [inline]

Allows to log the information about the instantiated trie type

Definition at line 134 of file c2d_hybrid_trie.hpp.

8.6.4.8 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType >::pre_allocate (const size_t *counts[LM_M_GRAM_LEVEL_MAX]*) [virtual]

This method can be used to provide the N-gram count information That should allow for pre-allocation of the memory For more details

See also

LayeredTrieBase

Definition at line 122 of file c2d_hybrid_trie.cpp.

8.6.4.9 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType >::set_def_unk_word_prob (const prob_weight *prob*)

See also

word_index_trie_base

Definition at line 84 of file c2d_hybrid_trie.cpp.

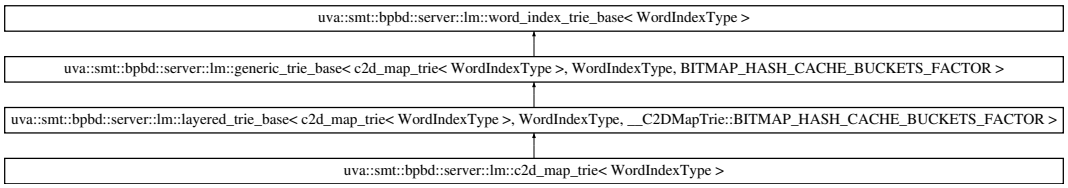
The documentation for this class was generated from the following files:

- inc/server/lm/models/c2d_hybrid_trie.hpp
- src/server/lm/models/c2d_hybrid_trie.cpp

8.7 uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType > Class Template Reference

`#include <c2d_map_trie.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType >:



Public Types

- typedef layered_trie_base< c2d_map_trie< WordIndexType >, WordIndexType, __C2DMapTrie::BITMAP↩ _HASH_CACHE_BUCKETS_FACTOR > BASE

Public Member Functions

- c2d_map_trie (WordIndexType &word_index, const float mgram_mem_factor=__C2DMapTrie::UM_M_GR↩ AM_MEMORY_FACTOR, const float ngram_mem_factor=__C2DMapTrie::UM_N_GRAM_MEMORY_FA↩ CTOR)
- float get_unk_word_prob () const
- bool get_ctx_id (const phrase_length level_idx, const TShortId word_id, TLongId &ctx_id) const
- void log_model_type_info () const
- virtual void pre_allocate (const size_t counts[LM_M_GRAM_LEVEL_MAX])
- template<phrase_length CURR_LEVEL>
  void add_m_gram (const model_m_gram &gram)
- void get_unigram_payload (m_gram_query &query) const
- void get_m_gram_payload (m_gram_query &query, MGramStatusEnum &status) const
- void get_n_gram_payload (m_gram_query &query, MGramStatusEnum &status) const
- void set_def_unk_word_prob (const prob_weight prob)
- virtual ~c2d_map_trie ()

Additional Inherited Members

8.7.1 Detailed Description

template<typename WordIndexType>class uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType >

This is a HashMpa based ATrie interface implementation class. Note 1: This implementation uses the unsigned long for the hashes it is not optimal Note 2: the unordered_map might be not as efficient as a hash_map with respect to memory usage but it is supposed to be faster

This implementation is chosen because it resembles the ordered array implementation from: "Faster and Smaller N -Gram Language Models" Adam Pauls Dan Klein Computer Science Division University of California, Berkeley

and unordered_maps showed good performance in: "Efficient in-memory data structures for n-grams indexing" D. Robenek, J. Platos. and V. Snásel Department of Computer Science, FEI, VSB – Technical University of Ostrava

1. listopadu 15, 708 33, Ostrava-Poruba, Czech Republic {daniel.robenek.st, jan.platos, vaclav.snasel}.cz

Definition at line 85 of file c2d_map_trie.hpp.

8.7.2 Member Typedef Documentation

8.7.2.1 template<typename WordIndexType > typedef layered_trie_base<c2d_map_trie<Word↩ IndexType>, WordIndexType, __C2DMapTrie::BITMAP_HASH_CACHE_BUCKETS_FACTOR> uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType >::BASE

Definition at line 87 of file c2d_map_trie.hpp.

8.7.3 Constructor & Destructor Documentation

8.7.3.1 template<typename WordIndexType > uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType >::c2d_map_trie (WordIndexType & *word_index*, const float *mgram_mem_factor* = `__C2DMapTrie::UM_M_GRAM_MEMORY_FACTOR`, const float *ngram_mem_factor* = `__C2DMapTrie::UM_N_GRAM_MEMORY_FACTOR`) `[explicit]`

The basic class constructor, accepts memory factors that are the coefficients used when pre-allocating memory for unordered maps.

If a factor is equal to 0.0 then no memory is pre-allocated. If the factor is equal to 1.0 then there is only as much preallocated as needed to store the gram entries. The latter is typically not enough as unordered_map needs more memory for internal administration. If there is not enough memory pre-allocated then additional allocations will take place but it does not alway lead to more efficient memory usage. The observed behavior is that it is better to pre-allocate a bit more memory beforehand, than needed. This leads to less memory consumption. Depending on the type of unordered_map key/value pair types the advised factor values are from 2.0 to 2.6. Because it can not be optimally determined beforehand, these are made constructor parameters so that they can be configured by the used. This breaks encapsulation a bit, exposing the internals, but there is no other better way, for fine tuning the memory usage.

Parameters

word_index	the word index to be used
mgram_mem_↩ factor	The M-Gram memory factor needed for the greedy allocator for the unordered_map
ngram_mem_↩ factor	The N-Gram memory factor needed for the greedy allocator for the unordered_map

Definition at line 48 of file c2d_map_trie.cpp.

8.7.3.2 template<typename WordIndexType > uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType >::~c2d_map_trie ( ) [virtual]

The basic destructor

Definition at line 133 of file c2d_map_trie.cpp.

8.7.4 Member Function Documentation

8.7.4.1 template<typename WordIndexType > template<phrase_length CURR_LEVEL> void uva::smt::bpbd↩ ::server::lm::c2d_map_trie< WordIndexType >::add_m_gram (const model_m_gram & *gram*) [inline]

Allows to retrieve the data storage structure for the M gram with the given M-gram level Id. M-gram context and last word Id. If the storage structure does not exist, return a new one. For more details

See also

LayeredTrieBase

Definition at line 158 of file c2d_map_trie.hpp.

8.7.4.2 template<typename WordIndexType > bool uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType >::get_ctx_id (const phrase_length *level_idx,* const TShortId *word_id,* TLongId & *ctx_id*) const [inline]

Computes the M-Gram context using the previous context and the current word id

See also

LayeredTrieBese

Definition at line 130 of file c2d_map_trie.hpp.

8.7.4.3 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType >::get_m_gram_payload (m_gram_query & *query,* MGramStatusEnum & *status*) const [inline]

Allows to retrieve the payload for the M-gram defined by the end word_id and ctx_id. For more details

See also

LayeredTrieBase

Definition at line 204 of file c2d_map_trie.hpp.

8.7.4.4 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType >::get_n_gram_payload (m_gram_query & *query*, MGramStatusEnum & *status*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for m==n

See also

GenericTrieBase

Definition at line 250 of file c2d_map_trie.hpp.

8.7.4.5 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType >::get_unigram_payload (m_gram_query & *query*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for m==1. The retrieval of a uni-gram data is always a success

See also

GenericTrieBase

Definition at line 189 of file c2d_map_trie.hpp.

8.7.4.6 template<typename WordIndexType > float uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType >::get_unk_word_prob ( ) const [inline]

Allows to retrieve the unknown target word log probability penalty

Returns

the target source word log probability penalty

Definition at line 122 of file c2d_map_trie.hpp.

8.7.4.7 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType >::log_model_type_info ( ) const [inline]

Allows to log the information about the instantiated trie type

Definition at line 140 of file c2d_map_trie.hpp.

8.7.4.8 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType >::pre_allocate (const size_t *counts[LM_M_GRAM_LEVEL_MAX]*) [virtual]

This method can be used to provide the N-gram count information That should allow for pre-allocation of the memory For more details

See also

LayeredTrieBase

Definition at line 117 of file c2d_map_trie.cpp.

8.7.4.9 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType >::set_def_unk_word_prob (const prob_weight *prob*)

See also

word_index_trie_base

Definition at line 86 of file c2d_map_trie.cpp.

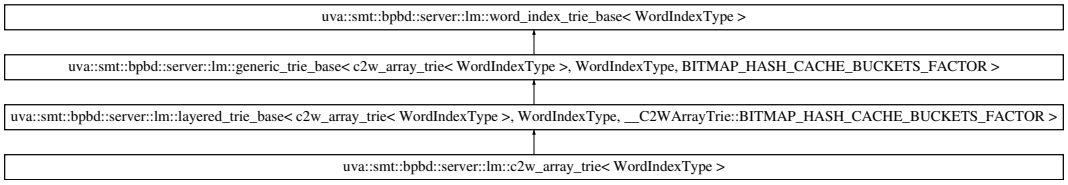
The documentation for this class was generated from the following files:

- inc/server/lm/models/c2d_map_trie.hpp
- src/server/lm/models/c2d_map_trie.cpp

8.8 uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType > Class Template Reference

`#include <c2w_array_trie.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >:



Classes

- struct TSubArrReference

Public Types

- typedef layered_trie_base< c2w_array_trie< WordIndexType >, WordIndexType, __C2WArrayTrie::BITM↩ AP_HASH_CACHE_BUCKETS_FACTOR > BASE

Public Member Functions

- c2w_array_trie (WordIndexType &p_word_index)
- float get_unk_word_prob () const
- bool get_ctx_id (const phrase_length level_idx, const TShortId word_id, TLongId &ctx_id) const
- void log_model_type_info () const
- virtual void pre_allocate (const size_t counts[LM_M_GRAM_LEVEL_MAX])
- void set_def_unk_word_prob (const prob_weight prob)
- template<phrase_length level>
  bool is_post_grams () const
- template<phrase_length CURR_LEVEL>
  void post_grams ()
- template<phrase_length CURR_LEVEL>
  void add_m_gram (const model_m_gram &gram)
- void get_unigram_payload (m_gram_query &query) const
- void get_m_gram_payload (m_gram_query &query, MGramStatusEnum &status) const
- void get_n_gram_payload (m_gram_query &query, MGramStatusEnum &status) const
- virtual ~c2w_array_trie ()

Protected Types

- typedef __C2WArrayTrie::TWordIdPBData TWordIdPBEntry
- typedef __C2WArrayTrie::TCtxIdProbData TCtxIdProbEntry

Protected Member Functions

- template<phrase_length CURR_LEVEL>
  void post_m_grams ()
- void post_n_grams ()

Additional Inherited Members

8.8.1 Detailed Description

template<typename WordIndexType>class uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >

This is the Context to word array memory trie implementation class.

WARNING: This trie assumes that the M-grams (1 <= M < N) are added to the Trie in an ordered way and there are no duplicates in the 1-Grams. The order is assumed to be lexicographical as in the ARPA files! This is also checked if the sanity checks are on see Globals.hpp!

Parameters

N	the maximum number of levels in the trie.

Definition at line 144 of file c2w_array_trie.hpp.

8.8.2 Member Typedef Documentation

8.8.2.1 template<typename WordIndexType > typedef layered_trie_base<c2w_array_trie<WordIndexType>, WordIndexType, __C2WArrayTrie::BITMAP_HASH_CACHE_BUCKETS_FACTOR> uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::BASE

Definition at line 146 of file c2w_array_trie.hpp.

8.8.2.2 template<typename WordIndexType > typedef __C2WArrayTrie::TCtxIdProbData uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::TCtxIdProbEntry [protected]

Definition at line 424 of file c2w_array_trie.hpp.

8.8.2.3 template<typename WordIndexType > typedef __C2WArrayTrie::TWordIdPBData uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::TWordIdPBEntry [protected]

Definition at line 423 of file c2w_array_trie.hpp.

8.8.3 Constructor & Destructor Documentation

8.8.3.1 template<typename WordIndexType > uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::c2w_array_trie (WordIndexType & *p_word_index*) [explicit]

The basic constructor

Parameters

p_word_index	the word index (dictionary) container

Definition at line 46 of file c2w_array_trie.cpp.

8.8.3.2 template<typename WordIndexType > uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::~c2w_array_trie ( ) [virtual]

The basic destructor

Definition at line 126 of file c2w_array_trie.cpp.

8.8.4 Member Function Documentation

8.8.4.1 template<typename WordIndexType > template<phrase_length CURR_LEVEL> void uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::add_m_gram (const model_m_gram & *gram*) [inline]

Allows to retrieve the data storage structure for the M gram with the given M-gram level Id. M-gram context and last word Id. If the storage structure does not exist, return a new one. For more details

See also

LayeredTrieBase

Definition at line 259 of file c2w_array_trie.hpp.

8.8.4.2 template<typename WordIndexType > bool uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::get_ctx_id (const phrase_length *level_idx,* const TShortId *word_id,* TLongId & *ctx_id*) const [inline]

Computes the M-Gram context using the previous context and the current word id

See also

LayeredTrieBese

Definition at line 166 of file c2w_array_trie.hpp.

8.8.4.3 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::get_m_gram_payload (m_gram_query & *query,* MGramStatusEnum & *status*) const [inline]

Allows to retrieve the payload for the M-gram defined by the end word_id and ctx_id. For more details

See also

LayeredTrieBase

Definition at line 328 of file c2w_array_trie.hpp.

8.8.4.4 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::get_n_gram_payload (m_gram_query & *query,* MGramStatusEnum & *status*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for m==n

See also

GenericTrieBase

Definition at line 366 of file c2w_array_trie.hpp.

8.8.4.5 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::get_unigram_payload (m_gram_query & *query*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for m==1. The retrieval of a uni-gram data is always a success

See also

GenericTrieBase

Definition at line 313 of file c2w_array_trie.hpp.

8.8.4.6 template<typename WordIndexType > float uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::get_unk_word_prob ( ) const [inline]

Allows to retrieve the unknown target word log probability penalty

Returns

the target source word log probability penalty

Definition at line 158 of file c2w_array_trie.hpp.

8.8.4.7 template<typename WordIndexType > template<phrase_length level> bool uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::is_post_grams ( ) const [inline]

This method allows to check if post processing should be called after all the X level grams are read. This method is virtual. For more details

See also

WordIndexTrieBase

Definition at line 224 of file c2w_array_trie.hpp.

8.8.4.8 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::log_model_type_info ( ) const [inline]

Allows to log the information about the instantiated trie type

Definition at line 202 of file c2w_array_trie.hpp.

8.8.4.9 template<typename WordIndexType > template<phrase_length CURR_LEVEL> void uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::post_grams ( ) [inline]

This method should be called after all the X level grams are read. For more details

See also

WordIndexTrieBase

Definition at line 236 of file c2w_array_trie.hpp.

8.8.4.10 template<typename WordIndexType > template<phrase_length CURR_LEVEL> void uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::post_m_grams ( ) [inline], [protected]

Definition at line 427 of file c2w_array_trie.hpp.

8.8.4.11 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::post_n_grams ( ) [inline], [protected]

Definition at line 447 of file c2w_array_trie.hpp.

8.8.4.12 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::pre_allocate (const size_t *counts[LM_M_GRAM_LEVEL_MAX]*) [virtual]

This method can be used to provide the N-gram count information That should allow for pre-allocation of the memory For more details

See also

LayeredTrieBase

Definition at line 65 of file c2w_array_trie.cpp.

8.8.4.13 template<typename WordIndexType > void uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::set_def_unk_word_prob (const prob_weight *prob*)

See also

word_index_trie_base

Definition at line 118 of file c2w_array_trie.cpp.

The documentation for this class was generated from the following files:

- inc/server/lm/models/c2w_array_trie.hpp
- src/server/lm/models/c2w_array_trie.cpp

8.9 uva::utils::containers::circular_queue< elem_type, capacity > Class Template Reference

`#include <circular_queue.hpp>`

Public Member Functions

- circular_queue ()
- circular_queue (const size_t num_elems, const elem_type *elems)
- circular_queue (const circular_queue &other, const size_t num_elems, const elem_type *elems)
- ~circular_queue ()
- string tail_to_string (const size_t num_elems) const
- bool is_equal_last (const circular_queue &other, const size_t num_elems) const
- void empty_queue ()
- size_t get_size () const
- const elem_type * get_elems () const
- size_t get_capacity () const
- void push_back (const elem_type &elem)
- void push_back (const size_t num_elems, const elem_type *elems)

8.9.1 Detailed Description

template<typename elem_type, size_t capacity>class uva::utils::containers::circular_queue< elem_type, capacity >

This class represents a circular queue class that is needed to store a limited and fixed amount of elements. This queue is designed to store only basic type elements WARNING: Class does a shallow copy of elements using the memcpy! So do not store here complex data structures with pointers and the overridden assign operator! WARNING: When comparing elements of the queue does a byte comparison with memcmp and not the == operator!

Definition at line 59 of file circular_queue.hpp.

8.9.2 Constructor & Destructor Documentation

8.9.2.1 template<typename elem_type , size_t capacity> uva::utils::containers::circular_queue< elem_type, capacity >::circular_queue ( ) [inline]

The basic constructor

Definition at line 65 of file circular_queue.hpp.

8.9.2.2 template<typename elem_type , size_t capacity> uva::utils::containers::circular_queue< elem_type, capacity >::circular_queue (const size_t *num_elems,* const elem_type * *elems*) [inline]

The basic constructor

Parameters

num_elems	the number of elements to put into the queue
elems	the elements to put into the queue

Definition at line 76 of file circular_queue.hpp.

8.9.2.3 template<typename elem_type , size_t capacity> uva::utils::containers::circular_queue< elem_type, capacity >::circular_queue (const circular_queue< elem_type, capacity > & *other,* const size_t *num_elems,* const elem_type * *elems*) [inline]

The special case of a copy constructor, it allows to take the parameter queue together with the extra elements and copy them into the given queue. This is done in a smart way to optimize performance.

Parameters

other	the other queue to copy from
num_elems	the number of extra elements
elems	the pointer to the array with the extra elements

Definition at line 95 of file circular_queue.hpp.

8.9.2.4 template<typename elem_type , size_t capacity> uva::utils::containers::circular_queue< elem_type, capacity >::~circular_queue ( ) [inline]

The basic destructor

Definition at line 139 of file circular_queue.hpp.

8.9.3 Member Function Documentation

8.9.3.1 template<typename elem_type , size_t capacity> void uva::utils::containers::circular_queue< elem_type, capacity >::empty_queue ( ) [inline]

Allows to empty the queue

Definition at line 187 of file circular_queue.hpp.

8.9.3.2 template<typename elem_type , size_t capacity> size_t uva::utils::containers::circular_queue< elem_type, capacity >::get_capacity ( ) const [inline]

Allows to obtain the maximum number of elements to store

Returns

the maximum number of elements to store

Definition at line 212 of file circular_queue.hpp.

8.9.3.3 template<typename elem_type , size_t capacity> const elem_type* uva::utils::containers::circular_queue< elem_type, capacity >::get_elems ( ) const [inline]

Allows to obtain the pointer to the array storing the elements

Returns

the pointer to the array storing the elements

Definition at line 204 of file circular_queue.hpp.

8.9.3.4 template<typename elem_type , size_t capacity> size_t uva::utils::containers::circular_queue< elem_type, capacity >::get_size ( ) const [inline]

Allows to obtain the number of stored elements

Returns

the number of stored elements

Definition at line 196 of file circular_queue.hpp.

8.9.3.5 template<typename elem_type , size_t capacity> bool uva::utils::containers::circular_queue< elem_type, capacity >::is_equal_last (const circular_queue< elem_type, capacity > & *other,* const size_t *num_elems*) const [inline]

Allows to check if a certain amount of this queue last elements is equal to the certain amount of the other queue last elements.

Parameters

other	the other queue to compare with
num_elems	the number of last elements to compare

Returns

true or false depending on? true if both queues have more than or equal to num_elems elements and the last num_elems are equal. true if both elements have less than num_elems but the number of elements is equal and the elements are equal false otherwise

Definition at line 167 of file circular_queue.hpp.

8.9.3.6 template<typename elem_type , size_t capacity> void uva::utils::containers::circular_queue< elem_type, capacity >::push_back (const elem_type & *elem*) [inline]

Allows to put the new element to the end of the queue, potentially pushing out the beginning of the queue element. The latter happens only if the maximum number of elements has been reached before this new element was pushed.

Parameters

elem	the element to be stored in the queue

Definition at line 223 of file circular_queue.hpp.

8.9.3.7 template<typename elem_type , size_t capacity> void uva::utils::containers::circular_queue< elem_type, capacity >::push_back (const size_t *num_elems*, const elem_type * *elems*) [inline]

Allows to push back an entire array

Definition at line 230 of file circular_queue.hpp.

8.9.3.8 template<typename elem_type , size_t capacity> string uva::utils::containers::circular_queue< elem_type, capacity >::tail_to_string (const size_t *num_elems*) const [inline]

Allows to get a string representation of the the specified number of tail elements. If there is less elements present in the queue then we represent as many as there are:

Parameters

num_elems	the number of tail elements to represent

Returns

the string of tail elements

Definition at line 149 of file circular_queue.hpp.

The documentation for this class was generated from the following file:

- inc/common/utils/containers/circular_queue.hpp

8.10 uva::smt::bpbd::client::client_config Struct Reference

`#include <client_config.hpp>`

Public Attributes

- string m_source_file
- string m_source_lang
- string m_target_file
- string m_target_lang
- string m_server
- uint16_t m_port
- uint64_t m_max_sent
- uint64_t m_min_sent
- bool is_pre_process

8.10.1 Detailed Description

This structure stores the translation client execution parameters

Definition at line 37 of file client_config.hpp.

8.10.2 Member Data Documentation

8.10.2.1 bool uva::smt::bpbd::client::client_config::is_pre_process

Definition at line 55 of file client_config.hpp.

8.10.2.2 uint64_t uva::smt::bpbd::client::client_config::m_max_sent

Definition at line 51 of file client_config.hpp.

8.10.2.3 uint64_t uva::smt::bpbd::client::client_config::m_min_sent

Definition at line 53 of file client_config.hpp.

8.10.2.4 uint16_t uva::smt::bpbd::client::client_config::m_port

Definition at line 49 of file client_config.hpp.

8.10.2.5 string uva::smt::bpbd::client::client_config::m_server

Definition at line 47 of file client_config.hpp.

8.10.2.6 string uva::smt::bpbd::client::client_config::m_source_file

Definition at line 39 of file client_config.hpp.

8.10.2.7 string uva::smt::bpbd::client::client_config::m_source_lang

Definition at line 41 of file client_config.hpp.

8.10.2.8 string uva::smt::bpbd::client::client_config::m_target_file

Definition at line 43 of file client_config.hpp.

8.10.2.9 string uva::smt::bpbd::client::client_config::m_target_lang

Definition at line 45 of file client_config.hpp.

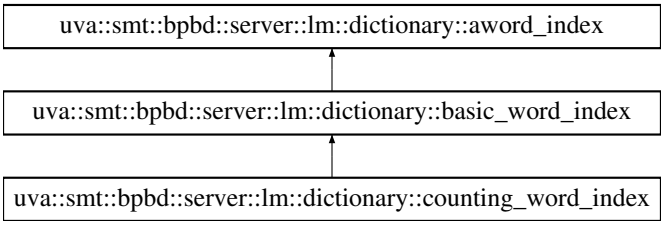
The documentation for this struct was generated from the following file:

- inc/client/client_config.hpp

8.11 uva::smt::bpbd::server::lm::dictionary::counting_word_index Class Reference

`#include <counting_word_index.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::dictionary::counting_word_index:



Public Member Functions

- counting_word_index (const float mem_factor)
- bool is_word_registering_needed () const
- word_uid register_word (const text_piece_reader &token)
- void count_word (const text_piece_reader &word, prob_weight prob)
- bool is_word_counts_needed () const
- void do_post_word_count ()
- bool is_post_actions_needed () const
- void do_post_actions ()

Static Public Member Functions

- static constexpr bool is_word_index_continuous ()

Additional Inherited Members

8.11.1 Detailed Description

This is a hash-map based implementation of the word index which extends the basic word index by word counting. This allows to count the word usages and then to issue lower word indexes to the more frequently used words. This allows for, for example, shorter M-gram ids.

Todo {Change or create a new version of the word index that will just use probabilities of the unigrams instead of counting words.}

Definition at line 83 of file counting_word_index.hpp.

8.11.2 Constructor & Destructor Documentation

8.11.2.1 uva::smt::bpbd::server::lm::dictionary::counting_word_index::counting_word_index (const float *mem_factor*) [inline]

The basic constructor

Parameters

mem_factor	the assigned memory factor for storage allocation in the unordered_map used for the word index

Definition at line 91 of file counting_word_index.hpp.

8.11.3 Member Function Documentation

8.11.3.1 void uva::smt::bpbd::server::lm::dictionary::counting_word_index::count_word (const text_piece_reader & *word*, prob_weight *prob*) [inline]

This method is to be used when the word counting is needed.

See also

AWordIndex

Definition at line 127 of file counting_word_index.hpp.

8.11.3.2 void uva::smt::bpbd::server::lm::dictionary::counting_word_index::do_post_actions ( ) [inline]

Is to be called if the post actions are needed right after that all the individual words have been added into the index.

See also

AWordIndex

Definition at line 230 of file counting_word_index.hpp.

8.11.3.3 void uva::smt::bpbd::server::lm::dictionary::counting_word_index::do_post_word_count ( ) [inline]

Should be called if the word count is needed after all the words have been counted.

See also

AWordIndex

Definition at line 151 of file counting_word_index.hpp.

8.11.3.4 bool uva::smt::bpbd::server::lm::dictionary::counting_word_index::is_post_actions_needed ( ) const [inline]

Indicates if the post-actions are needed. The post actions should be called after all the words have been filled into the index.

See also

AWordIndex

Definition at line 209 of file counting_word_index.hpp.

8.11.3.5 bool uva::smt::bpbd::server::lm::dictionary::counting_word_index::is_word_counts_needed ( ) const [inline]

This method allows to indicate whether word counting is needed by the given implementation of the word index.

See also

AWordIndex

Definition at line 141 of file counting_word_index.hpp.

8.11.3.6 static constexpr bool uva::smt::bpbd::server::lm::dictionary::counting_word_index::is_word_index_continuous ( ) [inline], [static]

Allows to indicate if the word index is continuous, i.e. it issues the word ids in a continuous range starting from 0.

See also

AWordIndex

Returns

true - this word index is continuous.

Definition at line 220 of file counting_word_index.hpp.

8.11.3.7 bool uva::smt::bpbd::server::lm::dictionary::counting_word_index::is_word_registering_needed ( ) const [inline]

This method allows to indicate whether registering a word is needed by the given implementation of the word index.

See also

AWordIndex

Definition at line 106 of file counting_word_index.hpp.

8.11.3.8 word_uid uva::smt::bpbd::server::lm::dictionary::counting_word_index::register_word (const text_piece_reader & *token*) [inline]

This function creates/gets a hash for the given word.

See also

AWordIndex

Definition at line 115 of file counting_word_index.hpp.

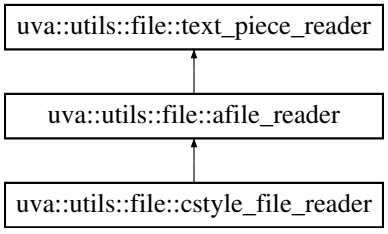
The documentation for this class was generated from the following file:

- inc/server/lm/dictionaries/counting_word_index.hpp

8.12 uva::utils::file::cstyle_file_reader Class Reference

`#include <cstyle_file_reader.hpp>`

Inheritance diagram for uva::utils::file::cstyle_file_reader:



Public Member Functions

- cstyle_file_reader (const char *fileName)
- cstyle_file_reader (const string &file_name)
- virtual void log_reader_type_info ()
- virtual ~cstyle_file_reader ()
- virtual void reset ()
- bool get_first_line (text_piece_reader &out)
- virtual bool is_open () const
- virtual operator bool () const
- virtual void close ()

Additional Inherited Members

8.12.1 Detailed Description

The file reader based on the simple C stream, should not use as much memory as MemoryMappedFileReader and potentially is faster than the C++ stream based reader.

Definition at line 51 of file cstyle_file_reader.hpp.

8.12.2 Constructor & Destructor Documentation

8.12.2.1 uva::utils::file::cstyle_file_reader::cstyle_file_reader (const char * *fileName*) [inline]

The basic constructor

Parameters

fileName	the file name

Definition at line 66 of file cstyle_file_reader.hpp.

8.12.2.2 uva::utils::file::cstyle_file_reader::cstyle_file_reader (const string & *file_name*) [inline]

The basic constructor

Parameters

file_name	the file name

Definition at line 87 of file cstyle_file_reader.hpp.

8.12.2.3 virtual uva::utils::file::cstyle_file_reader::~cstyle_file_reader ( ) [inline], [virtual]

Definition at line 97 of file cstyle_file_reader.hpp.

8.12.3 Member Function Documentation

8.12.3.1 virtual void uva::utils::file::cstyle_file_reader::close ( ) [inline], [virtual]

This method should be used to close the file

Reimplemented from uva::utils::file::afile_reader.

Definition at line 156 of file cstyle_file_reader.hpp.

8.12.3.2 bool uva::utils::file::cstyle_file_reader::get_first_line (text_piece_reader & *out*) [inline]

Definition at line 113 of file cstyle_file_reader.hpp.

8.12.3.3 virtual bool uva::utils::file::cstyle_file_reader::is_open ( ) const [inline], [virtual]

This method is used to check if the file was successfully opened.

Returns

true if the file is successfully opened otherwise false.

Implements uva::utils::file::afile_reader.

Definition at line 141 of file cstyle_file_reader.hpp.

8.12.3.4 virtual void uva::utils::file::cstyle_file_reader::log_reader_type_info ( ) [inline], [virtual]

Allows to log the information about the instantiated file reader type

Implements uva::utils::file::afile_reader.

Definition at line 93 of file cstyle_file_reader.hpp.

8.12.3.5 virtual uva::utils::file::cstyle_file_reader::operator bool ( ) const [inline], [virtual]

Checks if the file is present.

Returns

true if it is

Implements uva::utils::file::afile_reader.

Definition at line 149 of file cstyle_file_reader.hpp.

8.12.3.6 virtual void uva::utils::file::cstyle_file_reader::reset ( ) [inline], [virtual]

This method allows to reset the reading process and start reading the file from th first line again. The default implementation throws an exception.

Reimplemented from uva::utils::file::afile_reader.

Definition at line 107 of file cstyle_file_reader.hpp.

The documentation for this class was generated from the following file:

- inc/common/utils/file/cstyle_file_reader.hpp

8.13 uva::smt::bpbd::server::decoder::de_configurator Class Reference

`#include <de_configurator.hpp>`

Static Public Member Functions

- static void connect (const de_parameters &params)

- static void disconnect ()
- static sentence_decoder & allocate_decoder (acr_bool_flag is_stop, const string &source_sent, string &target_sent)
- static void dispose_decoder (sentence_decoder &dec)

8.13.1 Detailed Description

This class represents a singleton that allows to configure the decoding server that can create decoder instances. The interface is implemented as the configurations to the translation, reordering, and language models

Definition at line 48 of file de_configurator.hpp.

8.13.2 Member Function Documentation

8.13.2.1 static sentence_decoder& uva::smt::bpbd::server::decoder::de_configurator::allocate_decoder (acr_bool_flag *is_stop,* const string & *source_sent,* string & *target_sent*) [inline], [static]

Allows to get an instance of the decoder object.

Todo Pre-allocate decoders, make as many as there are threads

Parameters

is_stop	the flag that will be set to true in case one needs to abort the translation process.
source_sent	[in] the source language sentence to translate the source sentence is expected to be tokenized, reduced, and in the lower case.
target_sent	[out] the resulting target language sentence

Returns

an instance of the decoder object.

Definition at line 79 of file de_configurator.hpp.

8.13.2.2 static void uva::smt::bpbd::server::decoder::de_configurator::connect (const de_parameters & *params*) [inline], [static]

This method allows to "connect" to the decoder. The latter means configure it using the given data.

Parameters

params	the decoder parameters to be used, this class only stores the referent to the parameters.

Definition at line 57 of file de_configurator.hpp.

8.13.2.3 static void uva::smt::bpbd::server::decoder::de_configurator::disconnect ( ) [inline], [static]

Allows to disconnect from the decoder, i.e. clean up the memory etc.

Definition at line 64 of file de_configurator.hpp.

8.13.2.4 static void uva::smt::bpbd::server::decoder::de_configurator::dispose_decoder (sentence_decoder & *dec*) [inline], [static]

Allows to dispose the decoder

Todo Mark the decoder instance as available

Parameters

dec	the decoder to be returned

Definition at line 92 of file de_configurator.hpp.

The documentation for this class was generated from the following files:

- inc/server/decoder/de_configurator.hpp
- src/server/decoder/de_configurator.cpp

8.14 uva::smt::bpbd::server::decoder::de_parameters_struct Struct Reference

`#include <de_parameters.hpp>`

Public Member Functions

- de_parameters_struct ()
- de_parameters_struct & operator= (const de_parameters_struct &other)
- de_parameters_struct (const de_parameters_struct &other)
- void finalize ()

Public Attributes

- atomic< int32_t > m_distortion
- atomic< bool > m_is_dist
- atomic< uint32_t > m_ext_dist_left
- phrase_length m_max_s_phrase_len
- phrase_length m_max_t_phrase_len
- atomic< float > m_pruning_threshold
- atomic< uint32_t > m_stack_capacity
- atomic< float > m_word_penalty
- atomic< float > m_phrase_penalty
- atomic< uint32_t > m_num_best_trans
- atomic< bool > m_is_recombine

8.14.1 Detailed Description

This structure stores the decoder parameters

Definition at line 53 of file de_parameters.hpp.

8.14.2 Constructor & Destructor Documentation

8.14.2.1 uva::smt::bpbd::server::decoder::de_parameters_struct::de_parameters_struct ( ) [inline]

The basic constructor, does nothing

Definition at line 98 of file de_parameters.hpp.

8.14.2.2 uva::smt::bpbd::server::decoder::de_parameters_struct::de_parameters_struct (const de_parameters_struct & *other*) [inline]

The copy constructor

Parameters

other	the object to construct from

Definition at line 128 of file de_parameters.hpp.

8.14.3 Member Function Documentation

8.14.3.1 void uva::smt::bpbd::server::decoder::de_parameters_struct::finalize ( ) [inline]

Allows to verify the parameters to be correct.

Definition at line 135 of file de_parameters.hpp.

8.14.3.2 de_parameters_struct& uva::smt::bpbd::server::decoder::de_parameters_struct::operator= (const de_parameters_struct & *other*) [inline]

The assignment operator

Parameters

other	the object to assign from

Returns

this object updated with new values

Definition at line 106 of file de_parameters.hpp.

8.14.4 Member Data Documentation

8.14.4.1 atomic<int32_t> uva::smt::bpbd::server::decoder::de_parameters_struct::m_distortion

Definition at line 57 of file de_parameters.hpp.

8.14.4.2 atomic<uint32_t> uva::smt::bpbd::server::decoder::de_parameters_struct::m_ext_dist_left

Definition at line 68 of file de_parameters.hpp.

8.14.4.3 atomic<bool> uva::smt::bpbd::server::decoder::de_parameters_struct::m_is_dist

Definition at line 60 of file de_parameters.hpp.

8.14.4.4 atomic<bool> uva::smt::bpbd::server::decoder::de_parameters_struct::m_is_recombine

Definition at line 93 of file de_parameters.hpp.

8.14.4.5 phrase_length uva::smt::bpbd::server::decoder::de_parameters_struct::m_max_s_phrase_len

Definition at line 71 of file de_parameters.hpp.

8.14.4.6 phrase_length uva::smt::bpbd::server::decoder::de_parameters_struct::m_max_t_phrase_len

Definition at line 73 of file de_parameters.hpp.

8.14.4.7 atomic<uint32_t> uva::smt::bpbd::server::decoder::de_parameters_struct::m_num_best_trans

Definition at line 90 of file de_parameters.hpp.

8.14.4.8 atomic<float> uva::smt::bpbd::server::decoder::de_parameters_struct::m_phrase_penalty

Definition at line 84 of file de_parameters.hpp.

8.14.4.9 atomic<float> uva::smt::bpbd::server::decoder::de_parameters_struct::m_pruning_threshold

Definition at line 78 of file de_parameters.hpp.

8.14.4.10 atomic<uint32_t> uva::smt::bpbd::server::decoder::de_parameters_struct::m_stack_capacity

Definition at line 80 of file de_parameters.hpp.

8.14.4.11 atomic<float> uva::smt::bpbd::server::decoder::de_parameters_struct::m_word_penalty

Definition at line 82 of file de_parameters.hpp.

The documentation for this struct was generated from the following file:

- inc/server/decoder/de_parameters.hpp

8.15 uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR > Class Template Reference

`#include <dynamic_memory_arrays.hpp>`

Public Types

- typedef ELEMENT_TYPE TElemType
- typedef IDX_DATA_TYPE TIndexType
- typedef ELEMENT_TYPE * ELEMENT_TYPE_PTR

Public Member Functions

- dynamic_stack_array ()
- void pre_allocate (const IDX_DATA_TYPE capacity)
- ELEMENT_TYPE & allocate ()
- void shrink ()
- const ELEMENT_TYPE & operator[ ] (IDX_DATA_TYPE idx) const
- IDX_DATA_TYPE size () const
- const ELEMENT_TYPE * data () const
- bool has_data () const
- void sort ()
- void sort (typename T_IS_COMPARE_FUNC< ELEMENT_TYPE >::func_type is_less_func)
- ~dynamic_stack_array ()

Static Public Attributes

- static const size_t MAX_SIZE_TYPE_VALUE = MAX_U_TYPE_VALUES[sizeof (IDX_DATA_TYPE) - 1]
- static constexpr size_t PARAMETERS_SIZE_BYTES = (sizeof (ELEMENT_TYPE_PTR) + 2 * sizeof (IDX↩ _DATA_TYPE))

8.15.1 Detailed Description

template<typename ELEMENT_TYPE, typename IDX_DATA_TYPE, IDX_DATA_TYPE INITIAL_CAPACITY = 0, typename ELEME↩ NT_DEALLOC_FUNC< ELEMENT_TYPE >::func_ptr DESTRUCTOR = ELEMENT_DEALLOC_FUNC<ELEMENT_TYPE>::NULL↩ _FUNC_PTR>class uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DE↩ STRUCTOR >

This class represents a dynamic memory array and stores the main methods needed for its operation

Parameters

ELEMENT_TY↩ PE	the array element type
IDX_DATA_T↩ YPE	the type is to be used for the size, capacity and index variables, should be an unsigned type!
INITIAL_CAP↩ ACITY	the number of words, which defines the initial capacity.
DESTRUCTOR	the destructor function to be used on the elements when the container is deleted, default is NULL

Definition at line 201 of file dynamic_memory_arrays.hpp.

8.15.2 Member Typedef Documentation

8.15.2.1 template<typename ELEMENT_TYPE, typename IDX_DATA_TYPE, IDX_DATA_TYPE INITIAL_CAPACITY = 0, typename ELEMENT_DEALLOC_FUNC< ELEMENT_TYPE >::func_ptr DESTRUCTOR = ELEMENT_DEALLOC_FUNC<ELEMENT_TYPE>::NULL_FUNC_PTR> typedef ELEMENT_TYPE* uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >::ELEMENT_TYPE_PTR

Definition at line 211 of file dynamic_memory_arrays.hpp.

8.15.2.2 template<typename ELEMENT_TYPE, typename IDX_DATA_TYPE, IDX_DATA_TYPE INITIAL_CAPACITY = 0, typename ELEMENT_DEALLOC_FUNC< ELEMENT_TYPE >::func_ptr DESTRUCTOR = ELEMENT_DEALLOC_FUNC<ELEMENT_TYPE>::NULL_FUNC_PTR> typedef ELEMENT_TYPE uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >::TElemType

Definition at line 205 of file dynamic_memory_arrays.hpp.

8.15.2.3 template<typename ELEMENT_TYPE, typename IDX_DATA_TYPE, IDX_DATA_TYPE INITIAL_CAPACITY = 0, typename ELEMENT_DEALLOC_FUNC< ELEMENT_TYPE >::func_ptr DESTRUCTOR = ELEMENT_DEALLOC_FUNC<ELEMENT_TYPE>::NULL_FUNC_PTR> typedef IDX_DATA_TYPE uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >::TIndexType

Definition at line 208 of file dynamic_memory_arrays.hpp.

8.15.3 Constructor & Destructor Documentation

8.15.3.1 template<typename ELEMENT_TYPE, typename IDX_DATA_TYPE, IDX_DATA_TYPE INITIAL_CAPACITY = 0, typename ELEMENT_DEALLOC_FUNC< ELEMENT_TYPE >::func_ptr DESTRUCTOR = ELEMENT_DEALLOC_FUNC<ELEMENT_TYPE>::NULL_FUNC_PTR> uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >::dynamic_stack_array ( ) [inline]

The basic constructor, does not pre-allocate any memory

Definition at line 222 of file dynamic_memory_arrays.hpp.

8.15.3.2 template<typename ELEMENT_TYPE, typename IDX_DATA_TYPE, IDX_DATA_TYPE INITIAL_CAPACITY = 0, typename ELEMENT_DEALLOC_FUNC< ELEMENT_TYPE >::func_ptr DESTRUCTOR = ELEMENT_DEALLOC_FUNC<ELEMENT_TYPE>::NULL_FUNC_PTR> uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >::~dynamic_stack_array ( ) [inline]

The basic destructor

Definition at line 374 of file dynamic_memory_arrays.hpp.

8.15.4 Member Function Documentation

8.15.4.1 template<typename ELEMENT_TYPE, typename IDX_DATA_TYPE, IDX_DATA_TYPE INITIAL_CAPACITY = 0, typename ELEMENT_DEALLOC_FUNC< ELEMENT_TYPE >::func_ptr DESTRUCTOR = ELEMENT_DEALLOC_FUNC<ELEMENT_TYPE>::NULL_FUNC_PTR> ELEMENT_TYPE& uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >::allocate ( ) [inline]

Allows to retrieve the next new/unused element. Reallocates memory, if needed, to get space for the new element

Returns

the next new element

Definition at line 273 of file dynamic_memory_arrays.hpp.

8.15.4.2 template<typename ELEMENT_TYPE, typename IDX_DATA_TYPE, IDX_DATA_TYPE INITIAL_CAPACITY = 0, typename ELEMENT_DEALLOC_FUNC< ELEMENT_TYPE >::func_ptr DESTRUCTOR = ELEMENT_DEALLOC_FUNC<ELEMENT_TYPE>::NULL_FUNC_PTR> const ELEMENT_TYPE* uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >::data ( ) const [inline]

Allows to get the pointer to the stored data, note that this pointer is only guaranteed to be valid until a new element is added to the array, due to possible memory reallocation

Returns

the pointer to the data array

Definition at line 339 of file dynamic_memory_arrays.hpp.

8.15.4.3 template<typename ELEMENT_TYPE, typename IDX_DATA_TYPE, IDX_DATA_TYPE INITIAL_CAPACITY = 0, typename ELEMENT_DEALLOC_FUNC< ELEMENT_TYPE >::func_ptr DESTRUCTOR = ELEMENT_DEALLOC_FUNC<ELEMENT_TYPE>::NULL_FUNC_PTR> bool uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >::has_data ( ) const [inline]

Allows to check if there is data stored

Returns

true if there is at least one data element stored otherwise false

Definition at line 348 of file dynamic_memory_arrays.hpp.

8.15.4.4 template<typename ELEMENT_TYPE, typename IDX_DATA_TYPE, IDX_DATA_TYPE INITIAL_CAPACITY = 0, typename ELEMENT_DEALLOC_FUNC< ELEMENT_TYPE >::func_ptr DESTRUCTOR = ELEMENT_DEALLOC_FUNC<ELEMENT_TYPE>::NULL_FUNC_PTR> const ELEMENT_TYPE& uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >::operator[] (IDX_DATA_TYPE *idx*) const [inline]

This operator allows to retrieve the reference to an array element by the given index

Parameters

idx	the array element index

Returns

the reference to the array element under the given index

Exceptions

out_of_range	exception if the index is outside the array size.

Definition at line 310 of file dynamic_memory_arrays.hpp.

8.15.4.5 template<typename ELEMENT_TYPE, typename IDX_DATA_TYPE, IDX_DATA_TYPE INITIAL_CAPACITY = 0, typename ELEMENT_DEALLOC_FUNC< ELEMENT_TYPE >::func_ptr DESTRUCTOR = ELEMENT_DEALLOC_FUNC<ELEMENT_TYPE>::NULL_FUNC_PTR> void uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >::pre_allocate (const IDX_DATA_TYPE *capacity*) [inline]

Allows pre-allocate some capacity

Parameters

capacity	the capacity to pre-allocate

Definition at line 259 of file dynamic_memory_arrays.hpp.

8.15.4.6 template<typename ELEMENT_TYPE, typename IDX_DATA_TYPE, IDX_DATA_TYPE INITIAL_CAPACITY = 0, typename ELEMENT_DEALLOC_FUNC< ELEMENT_TYPE >::func_ptr DESTRUCTOR = ELEMENT_DEALLOC_FUNC<ELEMENT_TYPE>::NULL_FUNC_PTR> void uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >::shrink ( ) [inline]

De-allocated the un-used memory, if any

Definition at line 295 of file dynamic_memory_arrays.hpp.

8.15.4.7 template<typename ELEMENT_TYPE, typename IDX_DATA_TYPE, IDX_DATA_TYPE INITIAL_CAPACITY = 0, typename ELEMENT_DEALLOC_FUNC< ELEMENT_TYPE >::func_ptr DESTRUCTOR = ELEMENT_DEALLOC_FUNC<ELEMENT_TYPE>::NULL_FUNC_PTR> IDX_DATA_TYPE uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >::size ( ) const [inline]

Allows to retrieve the currently used number of elements

Returns

the number of elements stored in the stack array.

Definition at line 328 of file dynamic_memory_arrays.hpp.

8.15.4.8 template<typename ELEMENT_TYPE, typename IDX_DATA_TYPE, IDX_DATA_TYPE INITIAL_CAPACITY = 0, typename ELEMENT_DEALLOC_FUNC< ELEMENT_TYPE >::func_ptr DESTRUCTOR = ELEMENT_DEALLOC_FUNC<ELEME↩ NT_TYPE>::NULL_FUNC_PTR> void uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >::sort ( ) [inline]

Allows to sort the data stored in this stack array. How th data is sorted is defined by the < operator of the ELEM↩ ENT_TYPE

Definition at line 357 of file dynamic_memory_arrays.hpp.

8.15.4.9 template<typename ELEMENT_TYPE, typename IDX_DATA_TYPE, IDX_DATA_TYPE INITIAL_CAPACITY = 0, typename ELEMENT_DEALLOC_FUNC< ELEMENT_TYPE >::func_ptr DESTRUCTOR = ELEMENT_DEALLOC_FUNC<ELEME↩ NT_TYPE>::NULL_FUNC_PTR> void uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >::sort (typename T_IS_COMPARE_FUNC< ELEMENT_TYPE >::func_type *is_less_func*) [inline]

Allows to sort the data stored in this stack array. How th data is sorted is defined by the < operator of the ELEM↩ ENT_TYPE

Definition at line 366 of file dynamic_memory_arrays.hpp.

8.15.5 Member Data Documentation

8.15.5.1 template<typename ELEMENT_TYPE, typename IDX_DATA_TYPE, IDX_DATA_TYPE INITIAL_CAPACITY = 0, typename ELEMENT_DEALLOC_FUNC< ELEMENT_TYPE >::func_ptr DESTRUCTOR = ELEMENT_DEALLOC_FUN↩ C<ELEMENT_TYPE>::NULL_FUNC_PTR> const size_t uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >::MAX_SIZE_TYPE_VALUE = MAX_U_TYPE_VALUES[sizeof (IDX_DATA_TYPE) - 1] [static]

Definition at line 214 of file dynamic_memory_arrays.hpp.

8.15.5.2 template<typename ELEMENT_TYPE, typename IDX_DATA_TYPE, IDX_DATA_TYPE INITIAL_CAPACITY = 0, typename ELEMENT_DEALLOC_FUNC< ELEMENT_TYPE >::func_ptr DESTRUCTOR = ELEMENT_DEALLOC_FUNC<E↩ LEMENT_TYPE>::NULL_FUNC_PTR> constexpr size_t uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >::PARAMETERS_SIZE_BYTES = (sizeof (ELEMENT_TYPE_PTR) + 2 * sizeof (IDX_DATA_TYPE)) [static]

Definition at line 217 of file dynamic_memory_arrays.hpp.

The documentation for this class was generated from the following file:

- inc/common/utils/containers/dynamic_memory_arrays.hpp

8.16 uva::utils::containers::ELEMENT_DEALLOC_FUNC< ELEM_TYPE > Struct Template Reference

`#include <dynamic_memory_arrays.hpp>`

Public Types

- typedef std::function< void(ELEM_TYPE &) > func_type
- typedef void(* func_ptr) (ELEM_TYPE &)

Static Public Attributes

- static constexpr func_ptr NULL_FUNC_PTR = (typename ELEMENT_DEALLOC_FUNC<ELEM_TYPE>↩ ::func_ptr)NULL

8.16.1 Detailed Description

template<typename ELEM_TYPE>struct uva::utils::containers::ELEMENT_DEALLOC_FUNC< ELEM_TYPE >

The element deallocator function type for the ADynamicStackArray

Definition at line 186 of file dynamic_memory_arrays.hpp.

8.16.2 Member Typedef Documentation

8.16.2.1 template<typename ELEM_TYPE> typedef void(* uva::utils::containers::ELEMENT_DEALLOC_FUNC< ELEM_TYPE >::func_ptr) (ELEM_TYPE &)

Definition at line 188 of file dynamic_memory_arrays.hpp.

8.16.2.2 template<typename ELEM_TYPE> typedef std::function<void(ELEM_TYPE &) > uva::utils::containers::ELEMENT_DEALLOC_FUNC< ELEM_TYPE >::func_type

Definition at line 187 of file dynamic_memory_arrays.hpp.

8.16.3 Member Data Documentation

8.16.3.1 template<typename ELEM_TYPE> constexpr func_ptr uva::utils::containers↩ ::ELEMENT_DEALLOC_FUNC< ELEM_TYPE >::NULL_FUNC_PTR = (typename ELEMENT_DEALLOC_FUNC<ELEM_TYPE>::func_ptr)NULL [static]

Definition at line 189 of file dynamic_memory_arrays.hpp.

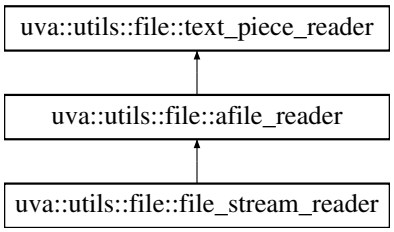
The documentation for this struct was generated from the following file:

- inc/common/utils/containers/dynamic_memory_arrays.hpp

8.17 uva::utils::file::file_stream_reader Class Reference

`#include <file_stream_reader.hpp>`

Inheritance diagram for uva::utils::file::file_stream_reader:



Public Member Functions

- file_stream_reader (const char *fileName)
- virtual void log_reader_type_info ()
- virtual ~file_stream_reader ()
- virtual void reset ()
- bool get_first_line (text_piece_reader &out)
- virtual bool is_open () const
- virtual operator bool () const
- virtual void close ()

Additional Inherited Members

8.17.1 Detailed Description

The file reader based on the simple ifstream, should not use as much memory as MemoryMappedFileReader and is seemingly as fast as the latter one on our applications.

Definition at line 49 of file file_stream_reader.hpp.

8.17.2 Constructor & Destructor Documentation

8.17.2.1 uva::utils::file::file_stream_reader::file_stream_reader (const char * *fileName*) [inline]

The basic constructor

Parameters

fileName	the file name

Definition at line 62 of file file_stream_reader.hpp.

8.17.2.2 virtual uva::utils::file::file_stream_reader::~file_stream_reader ( ) [inline], [virtual]

Definition at line 79 of file file_stream_reader.hpp.

8.17.3 Member Function Documentation

8.17.3.1 virtual void uva::utils::file::file_stream_reader::close ( ) [inline], [virtual]

This method should be used to close the file

Reimplemented from uva::utils::file::afile_reader.

Definition at line 138 of file file_stream_reader.hpp.

8.17.3.2 bool uva::utils::file::file_stream_reader::get_first_line (text_piece_reader & *out*) [inline]

Definition at line 91 of file file_stream_reader.hpp.

8.17.3.3 virtual bool uva::utils::file::file_stream_reader::is_open ( ) const [inline], [virtual]

This method is used to check if the file was successfully opened.

Returns

true if the file is successfully opened otherwise false.

Implements uva::utils::file::afile_reader.

Definition at line 123 of file file_stream_reader.hpp.

8.17.3.4 virtual void uva::utils::file::file_stream_reader::log_reader_type_info ( ) [inline], [virtual]

Allows to log the information about the instantiated file reader type

Implements uva::utils::file::afile_reader.

Definition at line 75 of file file_stream_reader.hpp.

8.17.3.5 virtual uva::utils::file::file_stream_reader::operator bool ( ) const [inline], [virtual]

Checks if the file is present.

Returns

true if it is

Implements uva::utils::file::afile_reader.

Definition at line 131 of file file_stream_reader.hpp.

8.17.3.6 virtual void uva::utils::file::file_stream_reader::reset ( ) [inline], [virtual]

This method allows to reset the reading process and start reading the file from th first line again. The default implementation throws an exception.

Reimplemented from uva::utils::file::afile_reader.

Definition at line 86 of file file_stream_reader.hpp.

The documentation for this class was generated from the following file:

- inc/common/utils/file/file_stream_reader.hpp

8.18 uva::utils::containers::fixed_size_hashmap< ELEMENT_TYPE, KEY_TYPE, IDX_TYPE > Class Template Reference

`#include <fixed_size_hashmap.hpp>`

Public Types

- typedef ELEMENT_TYPE TElemType

Public Member Functions

- fixed_size_hashmap (const double buckets_factor, const IDX_TYPE num_elems)
- ELEMENT_TYPE & add_new_element (const uint_fast64_t key_uid)
- ELEMENT_TYPE * get_element (const uint_fast64_t key_uid, const KEY_TYPE &key) const
- ~fixed_size_hashmap ()

Public Attributes

- const IDX_TYPE MAX_ELEMENT_INDEX

Static Public Attributes

- static constexpr IDX_TYPE NO_ELEMENT_INDEX = 0
- static constexpr IDX_TYPE MIN_ELEMENT_INDEX = NO_ELEMENT_INDEX + 1

8.18.1 Detailed Description

template<typename ELEMENT_TYPE, typename KEY_TYPE, typename IDX_TYPE = uint32_t>class uva::utils::containers::fixed_size_hashmap< ELEMENT_TYPE, KEY_TYPE, IDX_TYPE >

This class represents a fixed size hash map that stores a pre-defined number of elements. This is a linear probing hash map implementation, the linear probing hash map is currently known to be the fastest hash map there is, see: "Fast and Compact Hash Tables for Integer Keys" by Nikolas Askitis

Parameters

ELEMENT_TYPE	the element type, this type is expected to have the following interface: 1. operator==(const KEY_TYPE &); the comparison operator for the key value 2. static void clear(ELEMENT_TYPE &); the cleaning method to destroy contents of the element.
KEY_TYPE	the key type for retrieving the element
IDX_TYPE	the index type, is related to the number of elements

Definition at line 57 of file fixed_size_hashmap.hpp.

8.18.2 Member Typedef Documentation

8.18.2.1 template<typename ELEMENT_TYPE , typename KEY_TYPE , typename IDX_TYPE = uint32_t> typedef ELEMENT_TYPE uva::utils::containers::fixed_size_hashmap< ELEMENT_TYPE, KEY_TYPE, IDX_TYPE >::TElemType

Definition at line 59 of file fixed_size_hashmap.hpp.

8.18.3 Constructor & Destructor Documentation

8.18.3.1 template<typename ELEMENT_TYPE , typename KEY_TYPE , typename IDX_TYPE = uint32_t> uva::utils::containers::fixed_size_hashmap< ELEMENT_TYPE, KEY_TYPE, IDX_TYPE >::fixed_size_hashmap (const double *buckets_factor,* const IDX_TYPE *num_elems*) [inline], [explicit]

The basic constructor that allows to instantiate the map for the given number of elements. The number of buckets is computed based on the value: buckets_factor * (num_elems + 1) The latter is then rounded up to the next integer

being a power of two. The latter is needed to speed up the internal index computations.

Parameters

buckets_factor	the factor to compute the number of buckets from the number of elements
num_elems	the number of elements that will be stored in the map

Definition at line 77 of file fixed_size_hashmap.hpp.

8.18.3.2 template<typename ELEMENT_TYPE , typename KEY_TYPE , typename IDX_TYPE = uint32_t> uva::utils::containers::fixed_size_hashmap< ELEMENT_TYPE, KEY_TYPE, IDX_TYPE >::~fixed_size_hashmap ( ) [inline]

The basic destructor

Definition at line 172 of file fixed_size_hashmap.hpp.

8.18.4 Member Function Documentation

8.18.4.1 template<typename ELEMENT_TYPE , typename KEY_TYPE , typename IDX_TYPE = uint32_t> ELEMENT_TYPE& uva::utils::containers::fixed_size_hashmap< ELEMENT_TYPE, KEY_TYPE, IDX_TYPE >::add_new_element (const uint_fast64_t *key_uid*) [inline]

Allows to add a new element for the given hash value

Parameters

key_uid	the unique identifier representing the actual key value of the element. It can be e.g. a hash value of the key. Note that if one uses hash for a key uid then he or she has to accept the risk of collisions.

Returns

the reference to the new element

Definition at line 97 of file fixed_size_hashmap.hpp.

8.18.4.2 template<typename ELEMENT_TYPE , typename KEY_TYPE , typename IDX_TYPE = uint32_t> ELEMENT_TYPE* uva::utils::containers::fixed_size_hashmap< ELEMENT_TYPE, KEY_TYPE, IDX_TYPE >::get_element (const uint_fast64_t *key_uid,* const KEY_TYPE & *key*) const [inline]

Allows to retrieve the element for the given hash value and key

Parameters

key_uid	the unique identifier representing the actual key value of the element. It can be e.g. a hash value of the key. Note that if one uses hash for a key uid then he or she has to accept the risk of collisions.
key	the key value of the element

Returns

the pointer to the found element or NULL if nothing is found

Definition at line 144 of file fixed_size_hashmap.hpp.

8.18.5 Member Data Documentation

8.18.5.1 template<typename ELEMENT_TYPE , typename KEY_TYPE , typename IDX_TYPE = uint32_t> const IDX_TYPE uva::utils::containers::fixed_size_hashmap< ELEMENT_TYPE, KEY_TYPE, IDX_TYPE >::MAX_ELEMENT_INDEX

Definition at line 66 of file fixed_size_hashmap.hpp.

8.18.5.2 template<typename ELEMENT_TYPE , typename KEY_TYPE , typename IDX_TYPE = uint32_t> constexpr IDX_TYPE uva::utils::containers::fixed_size_hashmap< ELEMENT_TYPE, KEY_TYPE, IDX_TYPE >::MIN_ELEMENT_INDEX = NO_ELEMENT_INDEX + 1 [static]

Definition at line 64 of file fixed_size_hashmap.hpp.

8.18.5.3 template<typename ELEMENT_TYPE , typename KEY_TYPE , typename IDX_TYPE = uint32_t> constexpr IDX_TYPE uva::utils::containers::fixed_size_hashmap< ELEMENT_TYPE, KEY_TYPE, IDX_TYPE >::NO_ELEMENT_INDEX = 0 [static]

Definition at line 62 of file fixed_size_hashmap.hpp.

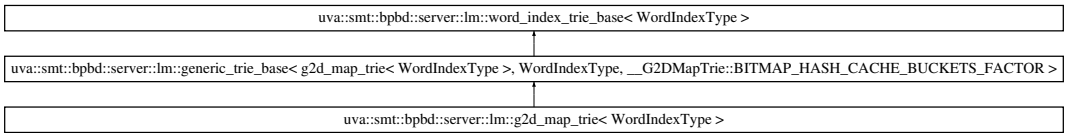
The documentation for this class was generated from the following file:

- inc/common/utils/containers/fixed_size_hashmap.hpp

8.19 uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType > Class Template Reference

`#include <g2d_map_trie.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType >:



Public Types

- typedef generic_trie_base< g2d_map_trie< WordIndexType >, WordIndexType, __G2DMapTrie::BITMAP↩ _HASH_CACHE_BUCKETS_FACTOR > BASE
- typedef __G2DMapTrie::S_M_GramData< m_gram_payload, word_uid > T_M_Gram_PB_Entry
- typedef __G2DMapTrie::S_M_GramData< prob_weight, word_uid > T_M_Gram_Prob_Entry

Public Member Functions

- g2d_map_trie (WordIndexType &word_index)
- float get_unk_word_prob () const
- void log_model_type_info () const
- void set_def_unk_word_prob (const prob_weight prob)
- virtual void pre_allocate (const size_t counts[LM_M_GRAM_LEVEL_MAX])
- template<phrase_length CURR_LEVEL>
  void add_m_gram (const model_m_gram &gram)
- void get_unigram_payload (m_gram_query &query) const
- void get_m_gram_payload (m_gram_query &query, MGramStatusEnum &status) const

- void get_n_gram_payload (m_gram_query &query, MGramStatusEnum &status) const
- virtual ~g2d_map_trie ()

Additional Inherited Members

8.19.1 Detailed Description

template<typename WordIndexType>class uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType >

This is a Gram to Data trie that is implemented as a HashMap.

Parameters

M_GRAM_LEVEL_MAX	- the maximum level of the considered N-gram, i.e. the N value

Definition at line 121 of file g2d_map_trie.hpp.

8.19.2 Member Typedef Documentation

8.19.2.1 template<typename WordIndexType > typedef generic_trie_base<g2d_map_trie<WordIndexType>, WordIndexType, __G2DMapTrie::BITMAP_HASH_CACHE_BUCKETS_FACTOR> uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType >::BASE

Definition at line 123 of file g2d_map_trie.hpp.

8.19.2.2 template<typename WordIndexType > typedef __G2DMapTrie::S_M_GramData<m_gram_payload, word_uid> uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType >::T_M_Gram_PB_Entry

Definition at line 124 of file g2d_map_trie.hpp.

8.19.2.3 template<typename WordIndexType > typedef __G2DMapTrie::S_M_GramData<prob_weight, word_uid> uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType >::T_M_Gram_Prob_Entry

Definition at line 125 of file g2d_map_trie.hpp.

8.19.3 Constructor & Destructor Documentation

8.19.3.1 template<typename WordIndexType > uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType >::g2d_map_trie (WordIndexType & *word_index*) [explicit]

The basic constructor

Parameters

_wordIndex	the word index to be used

Definition at line 48 of file g2d_map_trie.cpp.

8.19.3.2 template<typename WordIndexType > uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType >::~g2d_map_trie ( ) [virtual]

The basic class destructor

Definition at line 93 of file g2d_map_trie.cpp.

8.19.4 Member Function Documentation

8.19.4.1 template<typename WordIndexType > template<phrase_length CURR_LEVEL> void uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType >::add_m_gram (const model_m_gram & *gram*) [inline]

This method adds a M-Gram (word) to the trie where 1 < M < N

See also

GenericTrieBase

Definition at line 167 of file g2d_map_trie.hpp.

8.19.4.2 template<typename WordIndexType > void uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType >::get_m_gram_payload (m_gram_query & *query,* MGramStatusEnum & *status*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for 1<m<n

See also

GenericTrieBase

Parameters

query	the query containing the actual query data
status	the resulting status of the operation

Definition at line 219 of file g2d_map_trie.hpp.

8.19.4.3 template<typename WordIndexType > void uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType >::get_n_gram_payload (m_gram_query & *query,* MGramStatusEnum & *status*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for m==n

See also

GenericTrieBase

Parameters

query	the query containing the actual query data
status	the resulting status of the operation

Definition at line 238 of file g2d_map_trie.hpp.

8.19.4.4 template<typename WordIndexType > void uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType >::get_unigram_payload (m_gram_query & *query*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for m==1. The retrieval of a uni-gram data is always a success.

See also

GenericTrieBase

Definition at line 202 of file g2d_map_trie.hpp.

8.19.4.5 template<typename WordIndexType > float uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType >::get_unk_word_prob ( ) const [inline]

Allows to retrieve the unknown target word log probability penalty

Returns

the target source word log probability penalty

Definition at line 137 of file g2d_map_trie.hpp.

8.19.4.6 template<typename WordIndexType > void uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType >::log_model_type_info ( ) const [inline]

Allows to log the information about the instantiated trie type

Definition at line 144 of file g2d_map_trie.hpp.

8.19.4.7 template<typename WordIndexType > void uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType >::pre_allocate (const size_t *counts[LM_M_GRAM_LEVEL_MAX]*) [virtual]

This method can be used to provide the N-gram count information That should allow for pre-allocation of the memory

See also

GenericTrieBase

Definition at line 66 of file g2d_map_trie.cpp.

8.19.4.8 template<typename WordIndexType > void uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType >::set_def_unk_word_prob (const prob_weight *prob*)

See also

word_index_trie_base

Definition at line 85 of file g2d_map_trie.cpp.

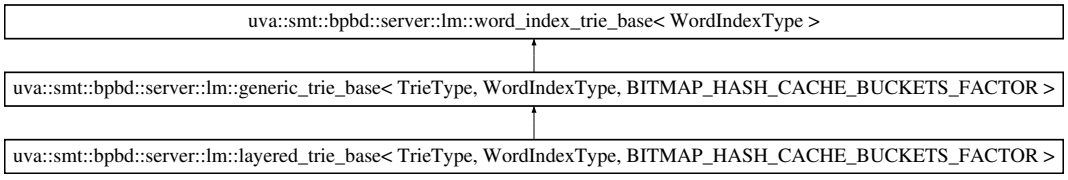
The documentation for this class was generated from the following files:

- inc/server/lm/models/g2d_map_trie.hpp
- src/server/lm/models/g2d_map_trie.cpp

8.20 uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR > Class Template Reference

`#include <generic_trie_base.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >:



Public Types

- typedef word_index_trie_base< WordIndexType > BASE

Public Member Functions

- generic_trie_base (WordIndexType &word_index)
- float get_unk_word_prob () const
- void pre_allocate (const size_t counts[LM_M_GRAM_LEVEL_MAX])
- template<phrase_length CURR_LEVEL>
  void add_m_gram (const model_m_gram &gram)
- void log_model_type_info () const
- void is_m_gram_potentially_present (m_gram_query &query, MGramStatusEnum &status) const
- void execute (m_gram_query &query) const
- void get_unigram_payload (m_gram_query &query, MGramStatusEnum &status) const
- void get_m_gram_payload (m_gram_query &query, MGramStatusEnum &status) const
- void get_n_gram_payload (m_gram_query &query, MGramStatusEnum &status) const
- void register_m_gram_cache (const model_m_gram &gram)
- virtual ~generic_trie_base ()

Static Public Member Functions

- static constexpr bool is_context_needed ()

Static Public Attributes

- static const bool NEEDS_BITMAP_HASH_CACHE = (BITMAP_HASH_CACHE_BUCKETS_FACTOR > 1)
- static const phrase_length MGRAM_IDX_OFFSET = 2
- static const phrase_length NUM_M_GRAM_LEVELS = LM_M_GRAM_LEVEL_MAX - MGRAM_IDX_OFFSET
- static const phrase_length NUM_M_N_GRAM_LEVELS = LM_M_GRAM_LEVEL_MAX - 1
- static const phrase_length N_GRAM_IDX_IN_M_N_ARR = LM_M_GRAM_LEVEL_MAX - MGRAM_IDX_OFFSET
- static const TShortId UNDEFINED_ARR_IDX = 0
- static const TShortId FIRST_VALID_CTX_ID = UNDEFINED_ARR_IDX + 1

Additional Inherited Members

8.20.1 Detailed Description

template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR>class uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >

This class defined the trie interface and functionality that is expected by the TrieDriver class

Definition at line 99 of file generic_trie_base.hpp.

8.20.2 Member Typedef Documentation

8.20.2.1 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> typedef word_index_trie_base<WordIndexType> uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::BASE

Definition at line 102 of file generic_trie_base.hpp.

8.20.3 Constructor & Destructor Documentation

8.20.3.1 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_C↩ ACHE_BUCKETS_FACTOR >::generic_trie_base (WordIndexType & *word_index*) [inline], [explicit]

The basic constructor

Parameters

word_index	the word index to be used

Definition at line 129 of file generic_trie_base.hpp.

8.20.3.2 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> virtual uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::~generic_trie_base ( ) [inline], [virtual]

The basic class destructor

Definition at line 346 of file generic_trie_base.hpp.

8.20.4 Member Function Documentation

8.20.4.1 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> template<phrase_length CURR_LEVEL> void uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::add_m_gram (const model_m_gram & *gram*) [inline]

This method adds a M-Gram (word) to the trie where $1 < M < N$

Parameters

gram	the M-Gram data

Exceptions

Exception	if the level of this M-gram is not such that $1 < M < N$

Definition at line 175 of file generic_trie_base.hpp.

8.20.4.2 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> void uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::execute (m_gram_query & *query*) const [inline]

This method allows to get the payloads and compute the (joint) m-gram probabilities.

Parameters

min_level	the minimum m-gram level to begin with
query	the query execution data for storing the query, and retrieved payloads, and resulting probabilities, and etc.

Definition at line 240 of file generic_trie_base.hpp.

8.20.4.3 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> void uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::get_m_gram_payload (m_gram_query & *query,* MGramStatusEnum & *status*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for 1<m<n

Parameters

query	the query containing the actual query data
status	the resulting status of the operation

Definition at line 313 of file generic_trie_base.hpp.

8.20.4.4 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> void uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::get_n_gram_payload (m_gram_query & *query*, MGramStatusEnum & *status*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for m==n

Parameters

query	the query containing the actual query data
status	the resulting status of the operation

Definition at line 322 of file generic_trie_base.hpp.

8.20.4.5 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> void uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::get_unigram_payload (m_gram_query & *query*, MGramStatusEnum & *status*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for m==1. The retrieval of a uni-gram data is always a success.

Parameters

query	the query containing the actual query data
status	the resulting status of the operation

Definition at line 304 of file generic_trie_base.hpp.

8.20.4.6 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> float uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::get_unk_word_prob ( ) const [inline]

Allows to retrieve the unknown target word log probability penalty

Returns

the target source word log probability penalty

Definition at line 142 of file generic_trie_base.hpp.

8.20.4.7 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> static constexpr bool uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::is_context_needed ( ) [inline], [static]

Allows to indicate whether the context id of an m-gram is to be computed while retrieving payloads

Returns

returns false, by default all generic tries need NO context ids when searching for data

Definition at line 150 of file generic_trie_base.hpp.

8.20.4.8 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> void uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::is_m_gram_potentially_present (m_gram_query & *query,* MGramStatusEnum & *status*) const [inline]

Allows to check if the given sub-m-gram, defined by the begin_word_idx and end_word_idx parameters, is potentially present in the trie. THis method must not be called for uni-grams, those always have a payload!

Parameters

query	the m-gram query data
status	[out] the resulting status of the operation

Definition at line 193 of file generic_trie_base.hpp.

8.20.4.9 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> void uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::log_model_type_info ( ) const [inline]

Allows to log the information about the instantiated trie type

Definition at line 182 of file generic_trie_base.hpp.

8.20.4.10 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> void uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::pre_allocate (const size_t *counts[LM_M_GRAM_LEVEL_MAX]*) [inline]

See also

WordIndexTrieBase

Definition at line 157 of file generic_trie_base.hpp.

8.20.4.11 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> void uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::register_m_gram_cache (const model_m_gram & *gram*) [inline]

Is to be used from the sub-classes from the add_X_gram methods. This method allows to register the given M-gram in internal high level caches if present.

WARNING: Is not to be used on uni-grams!!!

Parameters

gram	the M-gram to cache

Definition at line 335 of file generic_trie_base.hpp.

8.20.5 Member Data Documentation

8.20.5.1 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> const TShortId uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::FIRST_VALID_CTX_ID = UNDEFINED_ARR_IDX + 1 [static]

Definition at line 123 of file generic_trie_base.hpp.

8.20.5.2 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> const phrase_length uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::MGRAM_IDX_OFFSET = 2 [static]

Definition at line 108 of file generic_trie_base.hpp.

8.20.5.3 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> const phrase_length uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::N_GRAM_IDX_IN_M_N_ARR = LM_M_GRAM_LEVEL_MAX - MGRAM_IDX_OFFSET [static]

Definition at line 117 of file generic_trie_base.hpp.

8.20.5.4 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> const bool uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::NEEDS_BITMAP_HASH_CACHE = (BITMAP_HASH_CACHE_BUCKETS_FACTOR > 1) [static]

Definition at line 105 of file generic_trie_base.hpp.

8.20.5.5 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> const phrase_length uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::NUM_M_GRAM_LEVELS = LM_M_GRAM_LEVEL_MAX - MGRAM_IDX_OFFSET [static]

Definition at line 111 of file generic_trie_base.hpp.

8.20.5.6 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> const phrase_length uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::NUM_M_N_GRAM_LEVELS = LM_M_GRAM_LEVEL_MAX - 1 [static]

Definition at line 114 of file generic_trie_base.hpp.

8.20.5.7 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> const TShortId uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::UNDEFINED_ARR_IDX = 0 [static]

Definition at line 120 of file generic_trie_base.hpp.

The documentation for this class was generated from the following file:

- inc/server/lm/models/generic_trie_base.hpp

8.21 uva::utils::containers::alloc::greedy_memory_allocator< T > Class Template Reference

`#include <greedy_memory_allocator.hpp>`

Classes

- struct rebind

Public Types

- typedef T value_type
- typedef greedy_memory_storage::size_type size_type
- typedef std::ptrdiff_t difference_type

- typedef T * pointer
- typedef const T * const_pointer
- typedef T & reference
- typedef const T & const_reference

Public Member Functions

- greedy_memory_allocator (size_type numElems) throw ()
- greedy_memory_allocator (const greedy_memory_allocator &other) throw ()
- template<typename U >
  greedy_memory_allocator (const greedy_memory_allocator< U > &other) throw ()
- virtual ~greedy_memory_allocator () throw ()
- pointer address (reference obj) const
- const_pointer address (const_reference obj) const
- pointer allocate (size_type num, const_pointer cp=0)
- void deallocate (pointer ptr, size_type num)
- size_type available () const throw ()
- size_type max_size () const throw ()
- void construct (pointer ptr, const value_type &value)
- void destroy (pointer ptr)
- greedy_memory_storage & getStorageRef () const

Protected Attributes

- greedy_memory_storage & _manager

8.21.1 Detailed Description

template<typename T>class uva::utils::containers::alloc::greedy_memory_allocator< T >

This is the fixed memory allocator class for using in the tries. Here we pre-allocate some fixed size memory and then just give it out when needed. Since the Trie is build once and then is not changed, we do no do any memory deallocation here!

Definition at line 133 of file greedy_memory_allocator.hpp.

8.21.2 Member Typedef Documentation

8.21.2.1 template<typename T> typedef const T* uva::utils::containers::alloc::greedy_memory_allocator< T >::const_pointer

Definition at line 139 of file greedy_memory_allocator.hpp.

8.21.2.2 template<typename T> typedef const T& uva::utils::containers::alloc::greedy_memory_allocator< T >::const_reference

Definition at line 141 of file greedy_memory_allocator.hpp.

8.21.2.3 template<typename T> typedef std::ptrdiff_t uva::utils::containers::alloc::greedy_memory_allocator< T >::difference_type

Definition at line 137 of file greedy_memory_allocator.hpp.

8.21.2.4 template<typename T> typedef T* uva::utils::containers::alloc::greedy_memory_allocator< T >::pointer

Definition at line 138 of file greedy_memory_allocator.hpp.

8.21.2.5 template<typename T> typedef T& uva::utils::containers::alloc::greedy_memory_allocator< T >::reference

Definition at line 140 of file greedy_memory_allocator.hpp.

8.21.2.6 template<typename T> typedef greedy_memory_storage::size_type uva::utils::containers::alloc::greedy_memory_allocator< T >::size_type

Definition at line 136 of file greedy_memory_allocator.hpp.

8.21.2.7 template<typename T> typedef T uva::utils::containers::alloc::greedy_memory_allocator< T >::value_type

Definition at line 135 of file greedy_memory_allocator.hpp.

8.21.3 Constructor & Destructor Documentation

8.21.3.1 template<typename T> uva::utils::containers::alloc::greedy_memory_allocator< T >::greedy_memory_allocator (size_type *numElems*) throw) [inline]

The basic constructor.

Parameters

numElems	the number of elements of template type T to pre-allocate memory for.

Definition at line 152 of file greedy_memory_allocator.hpp.

8.21.3.2 template<typename T> uva::utils::containers::alloc::greedy_memory_allocator< T >::greedy_memory_allocator (const greedy_memory_allocator< T > & *other*) throw) [inline]

The basic copy constructor.

Definition at line 163 of file greedy_memory_allocator.hpp.

8.21.3.3 template<typename T> template<typename U > uva::utils::containers::alloc::greedy_memory_allocator< T >::greedy_memory_allocator (const greedy_memory_allocator< U > & *other*) throw) [inline]

The basic re-bind constructor. It is used internally by the container in case it needs to allocate other sort data than the stored container elements.

Definition at line 174 of file greedy_memory_allocator.hpp.

8.21.3.4 template<typename T> virtual uva::utils::containers::alloc::greedy_memory_allocator< T >::~greedy_memory_allocator ( ) throw) [inline], [virtual]

The standard destructor

Definition at line 182 of file greedy_memory_allocator.hpp.

8.21.4 Member Function Documentation

8.21.4.1 template<typename T> pointer uva::utils::containers::alloc::greedy_memory_allocator< T >::address (reference *obj*) const [inline]

Computes the address of the given object

Parameters

obj	the object to compute the pointer of

Returns

the computed pointer

Definition at line 192 of file greedy_memory_allocator.hpp.

8.21.4.2 template<typename T> const_pointer uva::utils::containers::alloc::greedy_memory_allocator< T >::address (const_reference *obj*) const [inline]

Computes the address of the given object

Parameters

obj	the object to compute the pointer of

Returns

the computed pointer

Definition at line 202 of file greedy_memory_allocator.hpp.

8.21.4.3 template<typename T> pointer uva::utils::containers::alloc::greedy_memory_allocator< T >::allocate (size_type *num,* const_pointer *cp* = 0) [inline]

Allocates memory for the given number of objects

Parameters

num	the number of objects to allocate
cp	NOT USED

Returns

the pointer to the first allocated object

Definition at line 213 of file greedy_memory_allocator.hpp.

8.21.4.4 template<typename T> size_type uva::utils::containers::alloc::greedy_memory_allocator< T >::available ( ) const throw) [inline]

Returns the available number of free elements we can store

Returns

the available number of free elements we can store

Definition at line 241 of file greedy_memory_allocator.hpp.

8.21.4.5 template<typename T> void uva::utils::containers::alloc::greedy_memory_allocator< T >::construct (pointer *ptr*, const value_type & *value*) [inline]

Calling the constructor on the given pointer

Parameters

ptr	the pointer to work with
value	the type value to work with

Definition at line 258 of file greedy_memory_allocator.hpp.

8.21.4.6 template<typename T> void uva::utils::containers::alloc::greedy_memory_allocator< T >::deallocate (pointer *ptr*, size_type *num*) [inline]

This function is supposed to deallocate the memory. We do not do that as this is fixed memory allocator

Parameters

ptr	the pointer to free memory from
num	the number of objects to deallocate

Definition at line 232 of file greedy_memory_allocator.hpp.

8.21.4.7 template<typename T> void uva::utils::containers::alloc::greedy_memory_allocator< T >::destroy (pointer *ptr*) [inline]

Calling the destructor on the given pointer

Parameters

ptr	the pointer to work with

Definition at line 267 of file greedy_memory_allocator.hpp.

8.21.4.8 template<typename T> greedy_memory_storage& uva::utils::containers::alloc::greedy_memory_allocator< T >::getStorageRef ( ) const [inline]

Returns the reference to the buffer manager

Returns

the reference to the buffer manager

Definition at line 276 of file greedy_memory_allocator.hpp.

8.21.4.9 template<typename T> size_type uva::utils::containers::alloc::greedy_memory_allocator< T >::max_size ( ) const throw) [inline]

Returns the maximum number of elements we can store

Returns

the maximum number of elements we can store

Definition at line 249 of file greedy_memory_allocator.hpp.

8.21.5 Member Data Documentation

8.21.5.1 template<typename T> greedy_memory_storage& uva::utils::containers::alloc::greedy_memory_allocator< T >::_manager [protected]

Definition at line 282 of file greedy_memory_allocator.hpp.

The documentation for this class was generated from the following file:

- inc/common/utils/containers/greedy_memory_allocator.hpp

8.22 uva::utils::containers::greedy_memory_storage Class Reference

`#include <greedy_memory_storage.hpp>`

Public Types

- typedef uint8_t TStorageData
- typedef std::size_t size_type

Public Member Functions

- greedy_memory_storage ()
- greedy_memory_storage (size_type numBytes)
- greedy_memory_storage (const greedy_memory_storage &source)
- ~greedy_memory_storage ()
- size_type getBufferSizeBytes () const
- size_type getAvailableBytes () const
- void * allocate (size_type num)

Protected Attributes

- void * _pBuffer
- vector< void * > _memoryBuffers
- size_type _numBytes
- size_type _allocBytes

8.22.1 Detailed Description

This is the greedy memory storage class that in the first place allocates some storage and then only grows it if more space is needed!

Definition at line 49 of file greedy_memory_storage.hpp.

8.22.2 Member Typedef Documentation

8.22.2.1 typedef std::size_t uva::utils::containers::greedy_memory_storage::size_type

Definition at line 56 of file greedy_memory_storage.hpp.

8.22.2.2 typedef uint8_t uva::utils::containers::greedy_memory_storage::TStorageData

Definition at line 53 of file greedy_memory_storage.hpp.

8.22.3 Constructor & Destructor Documentation

8.22.3.1 uva::utils::containers::greedy_memory_storage::greedy_memory_storage ( ) [inline],[explicit]

The basic constructor

Definition at line 61 of file greedy_memory_storage.hpp.

8.22.3.2 uva::utils::containers::greedy_memory_storage::greedy_memory_storage (size_type *numBytes*) [inline], [explicit]

The basic constructor of the greedy storage.

Parameters

numBytes	the number of bytes to pre-allocate the buffer for - the initial buffer capacity

Definition at line 68 of file greedy_memory_storage.hpp.

8.22.3.3 uva::utils::containers::greedy_memory_storage::greedy_memory_storage (const greedy_memory_storage & *source*) [inline]

The copy constructor

Definition at line 85 of file greedy_memory_storage.hpp.

8.22.3.4 uva::utils::containers::greedy_memory_storage::~greedy_memory_storage ( ) [inline]

The basic destructor.

Definition at line 95 of file greedy_memory_storage.hpp.

8.22.4 Member Function Documentation

8.22.4.1 void* uva::utils::containers::greedy_memory_storage::allocate (size_type *num*) [inline]

Allocates the memory of required size,if there is no enough space in the buffer, then reallocates!

Parameters

num	the number of bytes to allocate in the buffer

Returns

the pointer to the beginning of the allocated memory block

Definition at line 123 of file greedy_memory_storage.hpp.

8.22.4.2 size_type uva::utils::containers::greedy_memory_storage::getAvailableBytes ( ) const [inline]

Returns the number of free bytes remaining

Returns

the number of free bytes remaining

Definition at line 114 of file greedy_memory_storage.hpp.

8.22.4.3 size_type uva::utils::containers::greedy_memory_storage::getBufferSizeBytes ( ) const [inline]

Returns the current buffer size

Returns

the current buffer size

Definition at line 106 of file greedy_memory_storage.hpp.

8.22.5 Member Data Documentation

8.22.5.1 size_type uva::utils::containers::greedy_memory_storage::_allocBytes [protected]

Definition at line 152 of file greedy_memory_storage.hpp.

8.22.5.2 vector<void*> uva::utils::containers::greedy_memory_storage::_memoryBuffers [protected]

Definition at line 146 of file greedy_memory_storage.hpp.

8.22.5.3 size_type uva::utils::containers::greedy_memory_storage::_numBytes [protected]

Definition at line 149 of file greedy_memory_storage.hpp.

8.22.5.4 void* uva::utils::containers::greedy_memory_storage::_pBuffer [protected]

Definition at line 143 of file greedy_memory_storage.hpp.

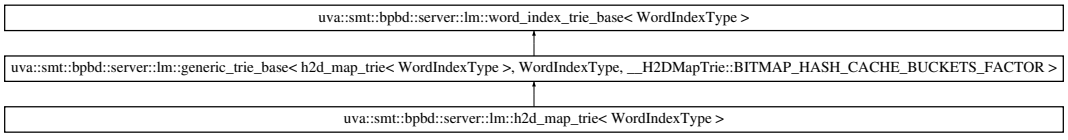
The documentation for this class was generated from the following file:

- inc/common/utils/containers/greedy_memory_storage.hpp

8.23 uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType > Class Template Reference

`#include <h2d_map_trie.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType >:



Public Types

- typedef generic_trie_base< h2d_map_trie< WordIndexType >, WordIndexType, __H2DMapTrie::BITMAP↩ _HASH_CACHE_BUCKETS_FACTOR > BASE
- typedef __H2DMapTrie::S_M_GramData< m_gram_payload > T_M_Gram_PB_Entry
- typedef __H2DMapTrie::S_M_GramData< prob_weight > T_M_Gram_Prob_Entry

Public Member Functions

- h2d_map_trie (WordIndexType &word_index)
- float get_unk_word_prob () const
- void log_model_type_info () const
- void set_def_unk_word_prob (const prob_weight prob)
- virtual void pre_allocate (const size_t counts[LM_M_GRAM_LEVEL_MAX])
- template<phrase_length CURR_LEVEL>
  void add_m_gram (const model_m_gram &gram)
- void get_unigram_payload (m_gram_query &query) const
- void get_m_gram_payload (m_gram_query &query, MGramStatusEnum &status) const
- void get_n_gram_payload (m_gram_query &query, MGramStatusEnum &status) const
- virtual ~h2d_map_trie ()

Additional Inherited Members

8.23.1 Detailed Description

template<typename WordIndexType>class uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType >

This is a Gram to Data trie that is implemented as a HashMap.

Parameters

M_GRAM_LE↩ VEL_MAX	- the maximum level of the considered N-gram, i.e. the N value

Definition at line 113 of file h2d_map_trie.hpp.

8.23.2 Member Typedef Documentation

8.23.2.1 template<typename WordIndexType> typedef generic_trie_base<h2d_map_trie<Word↩ IndexType>, WordIndexType, __H2DMapTrie::BITMAP_HASH_CACHE_BUCKETS_FACTOR> uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType >::BASE

Definition at line 115 of file h2d_map_trie.hpp.

8.23.2.2 template<typename WordIndexType> typedef __H2DMapTrie::S_M_GramData<m_gram_payload> uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType >::T_M_Gram_PB_Entry

Definition at line 116 of file h2d_map_trie.hpp.

8.23.2.3 template<typename WordIndexType> typedef __H2DMapTrie::S_M_GramData<prob_weight> uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType >::T_M_Gram_Prob_Entry

Definition at line 117 of file h2d_map_trie.hpp.

8.23.3 Constructor & Destructor Documentation

8.23.3.1 template<typename WordIndexType> uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType >::h2d_map_trie (WordIndexType & *word_index*) [explicit]

The basic constructor

Parameters

_wordIndex	the word index to be used

Definition at line 48 of file h2d_map_trie.cpp.

8.23.3.2 template<typename WordIndexType > uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType >::~h2d_map_trie ( ) [virtual]

The basic class destructor

Definition at line 88 of file h2d_map_trie.cpp.

8.23.4 Member Function Documentation

8.23.4.1 template<typename WordIndexType> template<phrase_length CURR_LEVEL> void uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType >::add_m_gram (const model_m_gram & *gram*) [inline]

This method adds a M-Gram (word) to the trie where 1 < M < N

See also

GenericTrieBase

Definition at line 159 of file h2d_map_trie.hpp.

8.23.4.2 template<typename WordIndexType> void uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType >::get_m_gram_payload (m_gram_query & *query,* MGramStatusEnum & *status*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for 1<m<n

See also

GenericTrieBase

Parameters

query	the query containing the actual query data
status	the resulting status of the operation

Definition at line 217 of file h2d_map_trie.hpp.

8.23.4.3 template<typename WordIndexType> void uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType >::get_n_gram_payload (m_gram_query & *query,* MGramStatusEnum & *status*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for m==n

See also

GenericTrieBase

Parameters

query	the query containing the actual query data
status	the resulting status of the operation

Definition at line 235 of file h2d_map_trie.hpp.

8.23.4.4 template<typename WordIndexType> void uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType >::get_unigram_payload (m_gram_query & *query*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for m==1. The retrieval of a uni-gram data is always a success.

See also

GenericTrieBase

Definition at line 201 of file h2d_map_trie.hpp.

8.23.4.5 template<typename WordIndexType> float uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType >::get_unk_word_prob ( ) const [inline]

Allows to retrieve the unknown target word log probability penalty

Returns

the target source word log probability penalty

Definition at line 129 of file h2d_map_trie.hpp.

8.23.4.6 template<typename WordIndexType> void uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType >::log_model_type_info ( ) const [inline]

Allows to log the information about the instantiated trie type

Definition at line 136 of file h2d_map_trie.hpp.

8.23.4.7 template<typename WordIndexType > void uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType >::pre_allocate (const size_t *counts[LM_M_GRAM_LEVEL_MAX]*) [virtual]

This method can be used to provide the N-gram count information That should allow for pre-allocation of the memory

See also

GenericTrieBase

Definition at line 67 of file h2d_map_trie.cpp.

8.23.4.8 template<typename WordIndexType > void uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType >::set_def_unk_word_prob (const prob_weight *prob*)

See also

word_index_trie_base

Definition at line 81 of file h2d_map_trie.cpp.

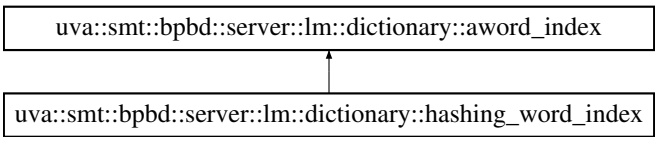
The documentation for this class was generated from the following files:

- inc/server/lm/models/h2d_map_trie.hpp
- src/server/lm/models/h2d_map_trie.cpp

8.24 uva::smt::bpbd::server::lm::dictionary::hashing_word_index Class Reference

`#include <hashing_word_index.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::dictionary::hashing_word_index:



Public Member Functions

- hashing_word_index (const float memory_factor)
- void reserve (const size_t num_words)
- size_t get_number_of_words (const size_t num_words) const
- word_uid get_word_id (const text_piece_reader &token) const
- bool is_word_registering_needed () const
- word_uid register_word (const text_piece_reader &token)
- bool is_word_counts_needed () const
- bool is_post_actions_needed () const
- virtual ~hashing_word_index ()

Static Public Member Functions

- static constexpr bool is_word_index_continuous ()

8.24.1 Detailed Description

This is a hashing word index, it is trivial - each word gets an id which is its hash value. This also means that any word is considered to be a known word. Therefore, in the Tries if the word id has no associated payload then an unknown word payload is to be used. Still the unknown and undefined word ids are reserved nd should not be issued.

Definition at line 58 of file hashing_word_index.hpp.

8.24.2 Constructor & Destructor Documentation

8.24.2.1 uva::smt::bpbd::server::lm::dictionary::hashing_word_index::hashing_word_index (const float *memory_factor*) [inline]

The basic constructor

Parameters

memory_factor	is not used, is here only for interface compliancy

Definition at line 65 of file hashing_word_index.hpp.

8.24.2.2 virtual uva::smt::bpbd::server::lm::dictionary::hashing_word_index::~hashing_word_index ( ) [inline], [virtual]

The basic destructor

Definition at line 140 of file hashing_word_index.hpp.

8.24.3 Member Function Documentation

8.24.3.1 size_t uva::smt::bpbd::server::lm::dictionary::hashing_word_index::get_number_of_words (const size_t *num_words*) const [inline]

See also

AWordIndex

Definition at line 80 of file hashing_word_index.hpp.

8.24.3.2 word_uid uva::smt::bpbd::server::lm::dictionary::hashing_word_index::get_word_id (const text_piece_reader & *token*) const [inline]

Does not detect unknown words. The returned word id is >= MIN_KNOWN_WORD_ID

See also

AWordIndex

Definition at line 89 of file hashing_word_index.hpp.

8.24.3.3 bool uva::smt::bpbd::server::lm::dictionary::hashing_word_index::is_post_actions_needed ( ) const [inline]

See also

AWordIndex

Definition at line 125 of file hashing_word_index.hpp.

8.24.3.4 bool uva::smt::bpbd::server::lm::dictionary::hashing_word_index::is_word_counts_needed ( ) const [inline]

See also

AWordIndex

Definition at line 118 of file hashing_word_index.hpp.

8.24.3.5 static constexpr bool uva::smt::bpbd::server::lm::dictionary::hashing_word_index::is_word_index_continuous ( ) [inline], [static]

See also

AWordIndex

Returns

false - this word index is not continuous.

Definition at line 133 of file hashing_word_index.hpp.

8.24.3.6 bool uva::smt::bpbd::server::lm::dictionary::hashing_word_index::is_word_registering_needed ( ) const [inline]

The returned word id is >= MIN_KNOWN_WORD_ID

See also

AWordIndex

Definition at line 103 of file hashing_word_index.hpp.

8.24.3.7 word_uid uva::smt::bpbd::server::lm::dictionary::hashing_word_index::register_word (const text_piece_reader & *token*) [inline]

The word registration is not needed, for this word index.

See also

AWordIndex

Definition at line 111 of file hashing_word_index.hpp.

8.24.3.8 void uva::smt::bpbd::server::lm::dictionary::hashing_word_index::reserve (const size_t *num_words*) [inline]

See also

AWordIndex

Definition at line 73 of file hashing_word_index.hpp.

The documentation for this class was generated from the following file:

- inc/server/lm/dictionaries/hashing_word_index.hpp

8.25 uva::smt::bpbd::common::messaging::id_manager< id_type > Class Template Reference

`#include <id_manager.hpp>`

Public Types

- typedef websocketpp::lib::lock_guard< websocketpp::lib::mutex > scoped_lock

Public Member Functions

- id_manager (const id_type min_id)
- uint32_t get_next_id ()
- const id_type & get_min_id () const

8.25.1 Detailed Description

template<typename id_type>class uva::smt::bpbd::common::messaging::id_manager< id_type >

This class is synchronized and an instance of the class is to be used in case one needs continuous ids to be issued in a multi-threaded environment.

Definition at line 43 of file id_manager.hpp.

8.25.2 Member Typedef Documentation

8.25.2.1 template<typename id_type> typedef websocketpp::lib::lock_guard<websocketpp::lib::mutex> uva::smt::bpbd::common::messaging::id_manager< id_type >::scoped_lock

Definition at line 45 of file id_manager.hpp.

8.25.3 Constructor & Destructor Documentation

8.25.3.1 template<typename id_type> uva::smt::bpbd::common::messaging::id_manager< id_type >::id_manager (const id_type *min_id*) [inline]

The basic class constructor for the id issuing entity. It is recommended to have the minimum value for the id higher that that allowed by the type itself. This way one can check for overflows.

Parameters

min_id	the minimum value of id

Definition at line 54 of file id_manager.hpp.

8.25.4 Member Function Documentation

8.25.4.1 template<typename id_type> const id_type& uva::smt::bpbd::common::messaging::id_manager< id_type >::get_min_id ( ) const [inline]

Allows to get the minimum value of the issued ids

Returns

the minimum value of the issued ids.

Definition at line 72 of file id_manager.hpp.

8.25.4.2 template<typename id_type> uint32_t uva::smt::bpbd::common::messaging::id_manager< id_type >::get_next_id ( ) [inline]

Allows to get the next id. This method is thread safe due to mutex locking.

Returns

the next id

Definition at line 62 of file id_manager.hpp.

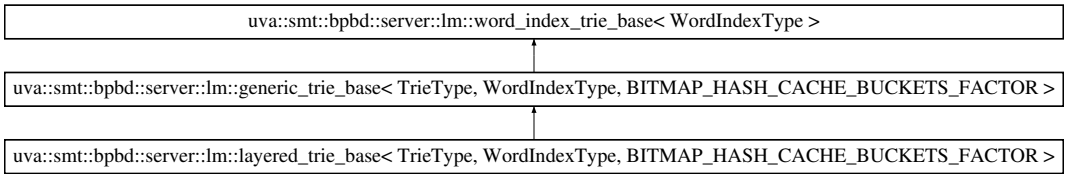
The documentation for this class was generated from the following file:

- inc/common/messaging/id_manager.hpp

8.26 uva::smt::bpbd::server::lm::layered_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR > Class Template Reference

`#include <layered_trie_base.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::layered_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >:



Public Types

- typedef generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR > BASE

Public Member Functions

- layered_trie_base (WordIndexType &word_index)
- void pre_allocate (const size_t counts[LM_M_GRAM_LEVEL_MAX])
- bool get_ctx_id (const phrase_length level_idx, const TShortId word_id, TLongId &ctx_id) const
- template<phrase_length CURR_LEVEL>
  bool get_cached_context_id (const model_m_gram &gram, TLongId &result) const
- template<phrase_length CURR_LEVEL>
  void set_cache_context_id (const model_m_gram &gram, TLongId &ctx_id)

Static Public Member Functions

- static constexpr bool is_context_needed ()

Protected Member Functions

- void ensure_context (m_gram_query &query, MGramStatusEnum &status) const

Additional Inherited Members

8.26.1 Detailed Description

template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR>class uva::smt::bpbd::server::lm::layered_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >

This class defined the trie interface and functionality that is expected by the TrieDriver class

Definition at line 153 of file layered_trie_base.hpp.

8.26.2 Member Typedef Documentation

8.26.2.1 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> typedef generic_trie_base<TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR> uva::smt::bpbd::server::lm::layered_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::BASE

Definition at line 156 of file layered_trie_base.hpp.

8.26.3 Constructor & Destructor Documentation

8.26.3.1 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> uva::smt::bpbd::server::lm::layered_trie_base< TrieType, WordIndexType, BITMAP_HASH_C↩ ACHE_BUCKETS_FACTOR >::layered_trie_base (WordIndexType & *word_index*) [inline], [explicit]

The basic constructor

Parameters

word_index	the word index to be used

Definition at line 162 of file layered_trie_base.hpp.

8.26.4 Member Function Documentation

8.26.4.1 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> void uva::smt::bpbd::server::lm::layered_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::ensure_context (m_gram_query & *query,* MGramStatusEnum & *status*) const [inline],[protected]

For the given query tries to ensure that the context is computed and stored. Also for the context the payload is retrieved. If the back-off is also not found sets its payload pointer to the zero payload structure. WARNING: This method is to be only called for minimal bi-gram queries! WARNING: Is only to be called if the context has not been computed yet

Parameters

query	the query to work with

Returns

true if the context was successfully computed, otherwise false.

Definition at line 241 of file layered_trie_base.hpp.

8.26.4.2 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> template<phrase_length CURR_LEVEL> bool uva::smt::bpbd::server::lm::layered_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::get_cached_context_id (const model_m_gram & *gram,* TLongId & *result*) const [inline]

Allows to retrieve the cached context id for the given M-gram if any

Parameters

mGram	the m-gram to get the context id for
result	the output parameter, will store the cached id, if any

Returns

true if there was nothing cached, otherwise false

Definition at line 202 of file layered_trie_base.hpp.

8.26.4.3 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> bool uva::smt::bpbd::server::lm::layered_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::get_ctx_id (const phrase_length *level_idx,* const TShortId *word_id,* TLongId & *ctx_id*) const [inline]

Allows to get the the new context id for the word and previous context id given the level

Parameters

level_idx	the m-gram level index, where m is > 1 and index is computed as m - 2;
word_id	the word id on this level
ctx_id	the previous level context id

Returns

true if computation of the next context is succeeded

Definition at line 191 of file layered_trie_base.hpp.

8.26.4.4 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> static constexpr bool uva::smt::bpbd::server::lm::layered_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::is_context_needed ( ) [inline], [static]

Allows to indicate whether the context id of an m-gram is to be computed while retrieving payloads

Returns

returns true, by default all layered tries need context ids when searching for data

Definition at line 173 of file layered_trie_base.hpp.

8.26.4.5 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> void uva::smt::bpbd::server::lm::layered_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::pre_allocate (const size_t *counts[LM_M_GRAM_LEVEL_MAX]*) [inline]

See also

GenericTrieBase

Definition at line 180 of file layered_trie_base.hpp.

8.26.4.6 template<typename TrieType, typename WordIndexType, uint8_t BITMAP_HASH_CACHE_BUCKETS_FACTOR> template<phrase_length CURR_LEVEL> void uva::smt::bpbd::server::lm::layered_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >::set_cache_context_id (const model_m_gram & *gram,* TLongId & *ctx_id*) [inline]

Allows to cache the context id of the given m-grams context

Parameters

gram	the m-gram to cache
ctx_id	the m-gram context id to cache.

Definition at line 221 of file layered_trie_base.hpp.

The documentation for this class was generated from the following file:

- inc/server/lm/models/layered_trie_base.hpp

8.27 uva::smt::bpbd::server::lm::arpa::lm_basic_builder< trie_type, reader_type > Class Template Reference

`#include <lm_basic_builder.hpp>`

Public Types

- typedef trie_type::WordIndexType WordIndexType

Public Member Functions

- lm_basic_builder (const lm_parameters &params, trie_type &trie, reader_type &file)
- void build ()
- virtual ~lm_basic_builder ()

8.27.1 Detailed Description

template<typename trie_type, typename reader_type>class uva::smt::bpbd::server::lm::arpa::lm_basic_builder< trie_type, reader_type >

This is the Trie builder class that reads an input file stream and creates n-grams and then records them into the provided Trie. This is an ARPA format based trie builder, so it expects that the provided model file contains a basic text model in ARPA format.

Definition at line 60 of file lm_basic_builder.hpp.

8.27.2 Member Typedef Documentation

8.27.2.1 template<typename trie_type, typename reader_type> typedef trie_type::WordIndexType uva::smt::bpbd::server::lm::arpa::lm_basic_builder< trie_type, reader_type >::WordIndexType

Definition at line 62 of file lm_basic_builder.hpp.

8.27.3 Constructor & Destructor Documentation

8.27.3.1 template<typename trie_type, typename reader_type> uva::smt::bpbd::server::lm::arpa::lm_basic_builder< trie_type, reader_type >::lm_basic_builder (const lm_parameters & *params*, trie_type & *trie*, reader_type & *file*)

The basic constructor that accepts a trie to be build up and the file stream to read from params the model parameters

Parameters

trie	the trie to fill in with data from the text corpus
_fstr	the file stream to read from

8.27.3.2 template<typename trie_type , typename reader_type > uva::smt::bpbd::server::lm::arpa::lm_basic_builder< trie_type, reader_type >::~lm_basic_builder ( ) [virtual]

Definition at line 68 of file lm_basic_builder.cpp.

8.27.4 Member Function Documentation

8.27.4.1 template<typename TrieType , typename TFileReaderModel > void uva::smt::bpbd::server::lm::arpa::lm_basic_builder< TrieType, TFileReaderModel >::build ( )

This function will read from the file and build the trie

Definition at line 455 of file lm_basic_builder.cpp.

The documentation for this class was generated from the following files:

- inc/server/lm/builders/lm_basic_builder.hpp
- src/server/lm/builders/lm_basic_builder.cpp

8.28 uva::smt::bpbd::server::lm::lm_configurator Class Reference

`#include <lm_configurator.hpp>`

Static Public Member Functions

- static void connect (const lm_parameters &params)
- static void disconnect ()
- static lm_slow_query_proxy & allocate_slow_query_proxy ()
- static void dispose_slow_query_proxy (lm_slow_query_proxy &query)
- static lm_fast_query_proxy & allocate_fast_query_proxy ()
- static void dispose_fast_query_proxy (lm_fast_query_proxy &query)

8.28.1 Detailed Description

This class represents a singleton that allows to configure the language model and then issues. query proxy objects for performing the queries against the internally encapsulated language model(s).

Definition at line 55 of file lm_configurator.hpp.

8.28.2 Member Function Documentation

8.28.2.1 static lm_fast_query_proxy& uva::smt::bpbd::server::lm::lm_configurator::allocate_fast_query_proxy ( ) [inline], [static]

Allows to return an instance of the query executor, is to be returned by calling the dispose method.

Returns

an instance of the query executor.

Definition at line 114 of file lm_configurator.hpp.

8.28.2.2 static lm_slow_query_proxy& uva::smt::bpbd::server::lm::lm_configurator::allocate_slow_query_proxy ( ) [inline], [static]

Allows to return an instance of the query executor, is to be returned by calling the dispose method.

Returns

an instance of the query executor.

Definition at line 94 of file lm_configurator.hpp.

8.28.2.3 static void uva::smt::bpbd::server::lm::lm_configurator::connect (const lm_parameters & *params*) [inline], [static]

This method allows to set the configuration parameters for the word index trie etc. This method is to be called only once! The latter is not checked but is a must.

Parameters

params	the language model parameters to be set, this class only stores the referent to the parameters.

Definition at line 65 of file lm_configurator.hpp.

8.28.2.4 static void uva::smt::bpbd::server::lm::lm_configurator::disconnect ( ) [inline], [static]

Allows to disconnect from the language model.

Definition at line 79 of file lm_configurator.hpp.

8.28.2.5 static void uva::smt::bpbd::server::lm::lm_configurator::dispose_fast_query_proxy (lm_fast_query_proxy & *query*) [inline], [static]

Dispose the previously allocated query object

Parameters

query	the query to dispose

Definition at line 125 of file lm_configurator.hpp.

8.28.2.6 static void uva::smt::bpbd::server::lm::lm_configurator::dispose_slow_query_proxy (lm_slow_query_proxy & *query*) [inline], [static]

Dispose the previously allocated query object

Parameters

query	the query to dispose

Definition at line 105 of file lm_configurator.hpp.

The documentation for this class was generated from the following files:

- inc/server/lm/lm_configurator.hpp
- src/server/lm/lm_configurator.cpp

8.29 uva::smt::bpbd::server::lm::__executor::lm_exec_params Struct Reference

`#include <lm_executor.hpp>`

Public Attributes

- lm_parameters m_lm_params
- string m_query_file_name

8.29.1 Detailed Description

This structure is needed to store the language model (query application) parameters

Definition at line 72 of file lm_executor.hpp.

8.29.2 Member Data Documentation

8.29.2.1 lm_parameters uva::smt::bpbd::server::lm::__executor::lm_exec_params::m_lm_params

Definition at line 74 of file lm_executor.hpp.

8.29.2.2 string uva::smt::bpbd::server::lm::__executor::lm_exec_params::m_query_file_name

Definition at line 77 of file lm_executor.hpp.

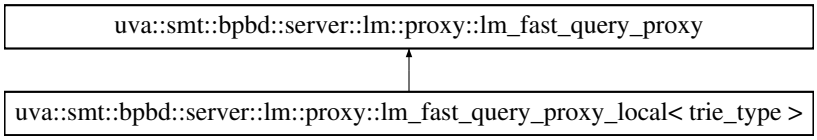
The documentation for this struct was generated from the following file:

- inc/server/lm/lm_executor.hpp

8.30 uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy Class Reference

`#include <lm_fast_query_proxy.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy:



Public Member Functions

- virtual ~lm_fast_query_proxy ()
- virtual prob_weight get_unk_word_prob () const =0
- virtual const word_uid & get_begin_tag_uid () const =0
- virtual const word_uid & get_end_tag_uid () const =0
- virtual void get_word_ids (text_piece_reader phrase, phrase_length &num_words, word_uid word_ids[tm::↩ TM_MAX_TARGET_PHRASE_LEN]) const =0
- virtual prob_weight execute (const phrase_length num_words, const word_uid *word_ids)=0
- virtual prob_weight execute (const phrase_length num_words, const word_uid *word_ids, phrase_length &min_level)=0

8.30.1 Detailed Description

This class represents a trie query proxy interface class. It allows to interact with templated trie queries in a uniform way.

Definition at line 48 of file lm_fast_query_proxy.hpp.

8.30.2 Constructor & Destructor Documentation

8.30.2.1 virtual uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy::~lm_fast_query_proxy ( ) [inline], [virtual]

The basic virtual destructor

Definition at line 54 of file lm_fast_query_proxy.hpp.

8.30.3 Member Function Documentation

8.30.3.1 virtual prob_weight uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy::execute (const phrase_length *num_words,* const word_uid * *word_ids*) [pure virtual]

Allows to execute m-gram the query. The query starts with the m-gram size one (1) and then grows until the maximum of LM_M_GRAM_LEVEL_MAX. After that m-grams of the LM_M_GRAM_LEVEL_MAX are computed via a sliding window: Let: "LM_MAX_QUERY_LEN = 4", "num_word_ids == 6" and "word_ids == w1w2w3w4w5w6" Then this method will compute the sum: P(w1) + P(w2|w1) + P(w3|w1w2) + P(w4|w1w2w3) + P(w5|w2w3w4) + P(w6|w3w4w5)

Parameters

num_words	stores the number of word ids, the maximum number of words must be LM_MAX_QUERY↩ _LEN
word_ids	the word identifiers of the words of the target phrase to compute the probability for

Returns

the probability weight for the given query

Implemented in uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >.

8.30.3.2 virtual prob_weight uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy::execute (const phrase_length *num_words,* const word_uid * *word_ids,* phrase_length & *min_level*) [pure virtual]

Allows to execute m-gram the query. The query starts with the m-gram size given by min_level and then grows until the maximum of LM_M_GRAM_LEVEL_MAX. After that m-grams of the LM_M_GRAM_LEVEL_MAX are computed via a sliding window: Let: "min_level == 2", "LM_MAX_QUERY_LEN = 4", "num_word_ids == 6" and "word_↩ ids == w1w2w3w4w5w6" Then this method will compute the sum: P(w2|w1) + P(w3|w1w2) + P(w4|w1w2w3) + P(w5|w2w3w4) + P(w6|w3w4w5)

Parameters

in	*num_words*	stores the number of word ids, the maximum number of words must be LM_↩ MAX_QUERY_LEN
in	*word_ids*	the word identifiers of the words of the target phrase to compute the probability for
	[in/out]	min_level the first m-gram level to consider, the next minimum m-gram level to consider, is limited by LM_M_GRAM_LEVEL_MAX

Returns

the resulting probability weight

Implemented in uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >.

8.30.3.3 virtual const word_uid& uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy::get_begin_tag_uid ( ) const [pure virtual]

Allows to retrieve the begin tag uid value

Returns

the begin tag "<s>" uid

Implemented in uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >.

8.30.3.4 virtual const word_uid& uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy::get_end_tag_uid ( ) const [pure virtual]

Allows to retrieve the end tag uid value

Returns

the end tag "</s>" uid

Implemented in uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >.

8.30.3.5 virtual prob_weight uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy::get_unk_word_prob ( ) const [pure virtual]

Allows to retrieve the unknown target word log probability penalty

Returns

the target source word log probability penalty

Implemented in uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >.

8.30.3.6 virtual void uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy::get_word_ids (text_piece_reader *phrase*, phrase_length & *num_words*, word_uid *word_ids[tm::TM_MAX_TARGET_PHRASE_LEN]*) const [pure virtual]

Allows to retrieve the target language phrase word ids. Note that the number of words in the target phrase should not exceed: TM_MAX_TARGET_PHRASE_LEN

Parameters

phrase	[in] the target language phrase
num_words	[out] the number of words to be set
word_ids	[out] the words ids to be set

Implemented in uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >.

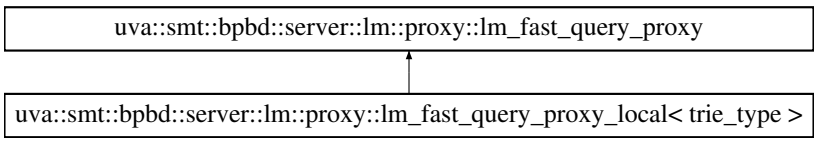
The documentation for this class was generated from the following file:

- inc/server/lm/proxy/lm_fast_query_proxy.hpp

8.31 uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type > Class Template Reference

`#include <lm_fast_query_proxy_local.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >:



Public Types

- typedef trie_type::WordIndexType word_index_type

Public Member Functions

- lm_fast_query_proxy_local (const trie_type &trie, const prob_weight &unk_word_prob, const word_uid &begin_tag_uid, const word_uid &end_tag_uid)
- virtual ~lm_fast_query_proxy_local ()
- virtual prob_weight get_unk_word_prob () const
- virtual const word_uid & get_begin_tag_uid () const
- virtual const word_uid & get_end_tag_uid () const
- virtual void get_word_ids (text_piece_reader phrase, phrase_length &num_words, word_uid word_ids[tm::↩ TM_MAX_TARGET_PHRASE_LEN]) const
- virtual prob_weight execute (const phrase_length num_words, const word_uid *word_ids)
- virtual prob_weight execute (const phrase_length num_words, const word_uid *word_ids, phrase_length &min_level)

Protected Member Functions

- string get_m_gram_str (const phrase_length begin_word_idx, const phrase_length end_word_idx) const
- string get_query_str () const
- void get_report_interm_results (const phrase_length begin_word_idx, const phrase_length first_end_word↩ _idx, const phrase_length last_end_word_idx)
- void report_final_result ()

8.31.1 Detailed Description

template<typename trie_type>class uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >

This is a local implementation of the language model query This implementation works with the local trie

Definition at line 51 of file lm_fast_query_proxy_local.hpp.

8.31.2 Member Typedef Documentation

8.31.2.1 template<typename trie_type> typedef trie_type::WordIndexType uva::smt::bpbd::server::lm::proxy::lm_↩ fast_query_proxy_local< trie_type >::word_index_type

Definition at line 54 of file lm_fast_query_proxy_local.hpp.

8.31.3 Constructor & Destructor Documentation

8.31.3.1 template<typename trie_type> uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >::lm_fast_query_proxy_local (const trie_type & *trie,* const prob_weight & *unk_word_prob,* const word_uid & *begin_tag_uid,* const word_uid & *end_tag_uid*) [inline]

The basic constructor that accepts the trie reference to query to Note that the begin and end tag uids are provided only for the sake of performance optimization.

Parameters

trie	the trie to query
unk_word_prob	the unknown word LM probability
begin_tag_uid	the begin sentence tag word uid

end_tag_uid	the begin sentence tag word uid

Definition at line 64 of file lm_fast_query_proxy_local.hpp.

8.31.3.2 template<typename trie_type> virtual uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >::~lm_fast_query_proxy_local ( ) [inline], [virtual]

See also

lm_fast_query_proxy

Definition at line 74 of file lm_fast_query_proxy_local.hpp.

8.31.4 Member Function Documentation

8.31.4.1 template<typename trie_type> virtual prob_weight uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >::execute (const phrase_length *num_words,* const word_uid * *word_ids*) [inline], [virtual]

See also

lm_query_proxy

Implements uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy.

Definition at line 133 of file lm_fast_query_proxy_local.hpp.

8.31.4.2 template<typename trie_type> virtual prob_weight uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >::execute (const phrase_length *num_words,* const word_uid * *word_ids,* phrase_length & *min_level*) [inline], [virtual]

See also

lm_query_proxy

Implements uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy.

Definition at line 149 of file lm_fast_query_proxy_local.hpp.

8.31.4.3 template<typename trie_type> virtual const word_uid& uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >::get_begin_tag_uid ( ) const [inline], [virtual]

See also

lm_query_proxy

Implements uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy.

Definition at line 88 of file lm_fast_query_proxy_local.hpp.

8.31.4.4 template<typename trie_type> virtual const word_uid& uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >::get_end_tag_uid ( ) const [inline], [virtual]

See also

lm_query_proxy

Implements uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy.

Definition at line 95 of file lm_fast_query_proxy_local.hpp.

8.31.4.5 template<typename trie_type> string uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >::get_m_gram_str (const phrase_length *begin_word_idx,* const phrase_length *end_word_idx*) const [inline], [protected]

For the given N-gram, for some level M <=N , this method allows to give the string of the object for which the probability is computed, e.g.: N-gram = "word1" -> result = "word1" N-gram = "word1 word2 word3" -> result = "word3 | word1 word2" for the first M tokens of the N-gram

Parameters

begin_word_idx	the m-gram's begin word index
end_word_idx	the m-gram's begin word index

Returns

the resulting string

Definition at line 224 of file lm_fast_query_proxy_local.hpp.

8.31.4.6 template<typename trie_type> string uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >::get_query_str ( ) const [inline], [protected]

For the given N-gram, this method allows to give the string of the object for which the probability is computed, e.g.: N-gram = "word1" -> result = "word1" N-gram = "word1 word2 word3" -> result = "word1 word2 word3"

Returns

the resulting string

Definition at line 247 of file lm_fast_query_proxy_local.hpp.

8.31.4.7 template<typename trie_type> void uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >::get_report_interm_results (const phrase_length *begin_word_idx,* const phrase_length *first_end_word_idx,* const phrase_length *last_end_word_idx*) [inline], [protected]

Allows add up the intermediate results of the loose sub-sub queries defined by the arguments

Parameters

begin_word_idx	the sub query begin word index
first_end_word_idx	the first sub-sub query end word index
last_end_word_idx	the last sub-sub query end word index

Definition at line 267 of file lm_fast_query_proxy_local.hpp.

8.31.4.8 template<typename trie_type> virtual prob_weight uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >::get_unk_word_prob ( ) const [inline], [virtual]

See also

lm_query_proxy

Implements uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy.

Definition at line 81 of file lm_fast_query_proxy_local.hpp.

8.31.4.9 template<typename trie_type> virtual void uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >::get_word_ids (text_piece_reader *phrase,* phrase_length & *num_words,* word_uid *word_ids[tm::TM_MAX_TARGET_PHRASE_LEN]*) const [inline], [virtual]

See also

lm_query_proxy

Implements uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy.

Definition at line 102 of file lm_fast_query_proxy_local.hpp.

8.31.4.10 template<typename trie_type> void uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >::report_final_result ( ) [inline], [protected]

Allows to report the total joint probability of the query

Definition at line 289 of file lm_fast_query_proxy_local.hpp.

The documentation for this class was generated from the following file:

- inc/server/lm/proxy/lm_fast_query_proxy_local.hpp

8.32 uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight > Class Template Reference

`#include <lm_gram_builder.hpp>`

Public Member Functions

- lm_gram_builder (const lm_parameters &params, WordIndexType &word_index, typename TAddGramFunct< WordIndexType >::func addGarmFunc)
- bool parse_line (text_piece_reader &data)
- virtual ~lm_gram_builder ()

Static Public Member Functions

- static bool unigram_to_prob (text_piece_reader &text, text_piece_reader &word, prob_weight &prob)

Protected Member Functions

- bool parse_to_gram (text_piece_reader &line)
- lm_gram_builder (const lm_gram_builder &orig)

Protected Attributes

- const lm_parameters & m_params
- WordIndexType & m_word_idx
- TAddGramFunct< WordIndexType >::func m_add_garm_func
- text_piece_reader m_token
- model_m_gram m_m_gram

Static Protected Attributes

- static const unsigned short int MIN_NUM_TOKENS_NGRAM_STR = 2
- static const unsigned short int MAX_NUM_TOKENS_NGRAM_STR = 3

8.32.1 Detailed Description

template<typename WordIndexType, phrase_length CURR_LEVEL, bool is_mult_weight>class uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight >

This class is responsible for splitting a piece of text in a number of ngrams and place it into the trie

Definition at line 58 of file lm_gram_builder.hpp.

8.32.2 Constructor & Destructor Documentation

8.32.2.1 template<typename WordIndexType , phrase_length CURR_LEVEL, bool is_mult_weight> uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight >::lm_gram_builder (const lm_parameters & *params*, WordIndexType & *word_index*, typename TAddGramFunct< WordIndexType >::func *addGarmFunc*)

The constructor to be used in order to instantiate a N-Gram builder

Parameters

word_index	the word index to be used
level	the level of the N-grams to be processed
addGarmFunc	the strategy for adding the N-grams

Definition at line 61 of file lm_gram_builder.cpp.

8.32.2.2 template<typename WordIndexType , phrase_length CURR_LEVEL, bool is_mult_weight> uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight >::~lm_gram_builder ( ) [virtual]

Definition at line 72 of file lm_gram_builder.cpp.

8.32.2.3 template<typename WordIndexType , phrase_length CURR_LEVEL, bool is_mult_weight> uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight >::lm_gram_builder (const lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight > & *orig*) [protected]

The copy constructor

Parameters

orig	the other builder to copy

Definition at line 67 of file lm_gram_builder.cpp.

8.32.3 Member Function Documentation

8.32.3.1 template<typename WordIndexType , phrase_length CURR_LEVEL, bool is_mult_weight> bool uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight >::parse_line (text_piece_reader & *data*)

This pure virtual method is supposed to parse the N-Gram string from the ARPA file format of a Back-Off language model and then add the obtained data to the Trie. This method has a default implementation that should work for N-grams with level > MIN_NGRAM_LEVEL and level < N

Parameters

data	the string to process, has to be space a separated sequence of tokens

Returns

returns true if the provided line is NOT recognized as the N-Gram of the specified level.

Definition at line 157 of file lm_gram_builder.cpp.

8.32.3.2 template<typename WordIndexType , phrase_length CURR_LEVEL, bool is_mult_weight> bool uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight >::parse_to_gram (text_piece_reader & *line*) [protected]

Parse the given text into a N-Gram entry from the ARPA file

Parameters

line	the piece of text to parse into the M-gram

Returns

true if the line has been successfully parsed

Definition at line 76 of file lm_gram_builder.cpp.

8.32.3.3 template<typename WordIndexType, phrase_length CURR_LEVEL, bool is_mult_weight> static bool uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight >::unigram_to_prob (text_piece_reader & *text,* text_piece_reader & *word,* prob_weight & *prob*) [inline], [static]

Takes the uni-gram line and parses it to the word and its probability, no back-off weight!

Parameters

text	the piece to read the uni-gram line from
word	[out] the text piece reader to read the word into
prob	[out] the variable to set the probability value into

Returns

true if the uni-gram was successfully parsed

Definition at line 89 of file lm_gram_builder.hpp.

8.32.4 Member Data Documentation

8.32.4.1 template<typename WordIndexType, phrase_length CURR_LEVEL, bool is_mult_weight> TAddGramFunct<WordIndexType>::func uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight >::m_add_garm_func [protected]

Definition at line 129 of file lm_gram_builder.hpp.

8.32.4.2 template<typename WordIndexType, phrase_length CURR_LEVEL, bool is_mult_weight> model_m_gram uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight >::m_m_gram [protected]

Definition at line 135 of file lm_gram_builder.hpp.

8.32.4.3 template<typename WordIndexType, phrase_length CURR_LEVEL, bool is_mult_weight> const lm_parameters& uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight >::m_params [protected]

Definition at line 124 of file lm_gram_builder.hpp.

8.32.4.4 template<typename WordIndexType, phrase_length CURR_LEVEL, bool is_mult_weight> text_piece_reader uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight >::m_token [protected]

Definition at line 132 of file lm_gram_builder.hpp.

8.32.4.5 template<typename WordIndexType, phrase_length CURR_LEVEL, bool is_mult_weight> WordIndexType& uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight >::m_word_idx [protected]

Definition at line 126 of file lm_gram_builder.hpp.

8.32.4.6 template<typename WordIndexType, phrase_length CURR_LEVEL, bool is_mult_weight> const unsigned short int uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight >::MAX_NUM_TOKENS_NGRAM_STR = 3 [static], [protected]

Definition at line 139 of file lm_gram_builder.hpp.

8.32.4.7 template<typename WordIndexType, phrase_length CURR_LEVEL, bool is_mult_weight> const unsigned short int uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight >::MIN_NUM_TOKENS_NGRAM_STR = 2 [static], [protected]

Definition at line 138 of file lm_gram_builder.hpp.

The documentation for this class was generated from the following files:

- inc/server/lm/builders/lm_gram_builder.hpp
- src/server/lm/builders/lm_gram_builder.cpp

8.33 uva::smt::bpbd::server::lm::arpa::lm_gram_builder_factory< TrieType > Class Template Reference

`#include <lm_gram_builder_factory.hpp>`

Public Types

- typedef TrieType::WordIndexType WordIndexType

Public Member Functions

- virtual ~lm_gram_builder_factory ()

Static Public Member Functions

- template<phrase_length CURR_LEVEL, bool is_mult_weight>
  static void get_builder (const lm_parameters &params, TrieType &trie, lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight > **ppBuilder)

8.33.1 Detailed Description

template<typename TrieType>class uva::smt::bpbd::server::lm::arpa::lm_gram_builder_factory< TrieType >

This is the ARPA N-gram Builder Factory class that is supposed to be used to instantiate the proper ARPA N-Gram Builder class. Note that there can be a small difference in the data provided for the N-grams of different levels. For example the N-gram of the maximum level does not have back-off weights, so knowing that can allow for a more optimal reading the data and filling in the Trie. Also the first level N-grams (N==1) are just words and have to be added as vocabulary words into the Trie and not as regular N-grams.

Definition at line 64 of file lm_gram_builder_factory.hpp.

8.33.2 Member Typedef Documentation

8.33.2.1 template<typename TrieType> typedef TrieType::WordIndexType uva::smt::bpbd::server::lm::arpa::lm_gram_builder_factory< TrieType >::WordIndexType

Definition at line 66 of file lm_gram_builder_factory.hpp.

8.33.3 Constructor & Destructor Documentation

8.33.3.1 template<typename TrieType> virtual uva::smt::bpbd::server::lm::arpa::lm_gram_builder_factory< TrieType >::~lm_gram_builder_factory ( ) [inline], [virtual]

Definition at line 112 of file lm_gram_builder_factory.hpp.

8.33.4 Member Function Documentation

8.33.4.1 template<typename TrieType> template<phrase_length CURR_LEVEL, bool is_mult_weight> static void uva::smt::bpbd::server::lm::arpa::lm_gram_builder_factory< TrieType >::get_builder (const lm_parameters & *params,* TrieType & *trie,* lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight > ** *ppBuilder*) [inline], [static]

This is a template method for getting the proper ARPA N-gram Builder for the given N-gram level. The two template parameters of this method N and doCache have to do with the Trie template parameters. N is the maximum N-gram level and doCache indicates whether the given Trie does caching of query results.

Note: the returned pointer to the dynamically allocated builder is to be freed by the caller!

Parameters

CURR_LEVEL	the level of the N-gram we currently need the builder for.
params	the model parameters weights are to be multiplies with the language model m-gram weight
trie	the trie to be filled in with the N-grams
pBuilder	the pointer to a dynamically allocated N-Gram builder

Definition at line 86 of file lm_gram_builder_factory.hpp.

The documentation for this class was generated from the following file:

- inc/server/lm/builders/lm_gram_builder_factory.hpp

8.34 uva::smt::bpbd::server::lm::lm_parameters Struct Reference

`#include <lm_parameters.hpp>`

Public Member Functions

- bool is_lm_weight () const
- const float & get_lm_weight () const
- void finalize ()

Public Attributes

- string m_conn_string
- size_t m_num_lambdas
- float m_lambdas [NUM_LM_FEATURES]

8.34.1 Detailed Description

This structure is needed to store the language model parameters

Definition at line 53 of file lm_parameters.hpp.

8.34.2 Member Function Documentation

8.34.2.1 void uva::smt::bpbd::server::lm::lm_parameters::finalize ( ) [inline]

Allows to verify the parameters to be correct.

Definition at line 83 of file lm_parameters.hpp.

8.34.2.2 const float& uva::smt::bpbd::server::lm::lm_parameters::get_lm_weight ( ) const [inline]

Allows to retrieve the language model m-gram weight

Returns

the language model m-gram weight

Definition at line 76 of file lm_parameters.hpp.

8.34.2.3 bool uva::smt::bpbd::server::lm::lm_parameters::is_lm_weight ( ) const [inline]

Allows to detect that the lm weight is set and needs to be used

Returns

true if we need to multiply with the lambda weight otherwise not.

Definition at line 68 of file lm_parameters.hpp.

8.34.3 Member Data Documentation

8.34.3.1 string uva::smt::bpbd::server::lm::lm_parameters::m_conn_string

Definition at line 55 of file lm_parameters.hpp.

8.34.3.2 float uva::smt::bpbd::server::lm::lm_parameters::m_lambdas[NUM_LM_FEATURES]

Definition at line 62 of file lm_parameters.hpp.

8.34.3.3 size_t uva::smt::bpbd::server::lm::lm_parameters::m_num_lambdas

Definition at line 59 of file lm_parameters.hpp.

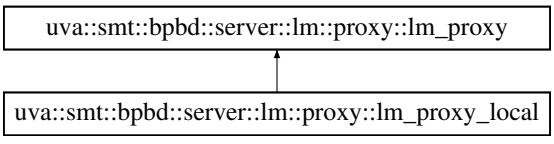
The documentation for this struct was generated from the following file:

- inc/server/lm/lm_parameters.hpp

8.35 uva::smt::bpbd::server::lm::proxy::lm_proxy Class Reference

`#include <lm_proxy.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::proxy::lm_proxy:



Public Member Functions

- virtual void connect (const lm_parameters &params)=0
- virtual void disconnect ()=0
- virtual ~lm_proxy ()
- virtual lm_slow_query_proxy & allocate_slow_query_proxy ()=0
- virtual void dispose_slow_query_proxy (lm_slow_query_proxy &query)=0
- virtual lm_fast_query_proxy & allocate_fast_query_proxy ()=0
- virtual void dispose_fast_query_proxy (lm_fast_query_proxy &query)=0

8.35.1 Detailed Description

This is the trie proxy interface class it allows to interact with templated tries in a uniform way.

Definition at line 44 of file lm_proxy.hpp.

8.35.2 Constructor & Destructor Documentation

8.35.2.1 virtual uva::smt::bpbd::server::lm::proxy::lm_proxy::~lm_proxy ( ) [inline], [virtual]

The basic virtual destructor

Definition at line 61 of file lm_proxy.hpp.

8.35.3 Member Function Documentation

8.35.3.1 virtual lm_fast_query_proxy& uva::smt::bpbd::server::lm::proxy::lm_proxy::allocate_fast_query_proxy ( ) [pure virtual]

This method allows to get a trie query executor for the given trie

Returns

the trie query proxy object

Implemented in uva::smt::bpbd::server::lm::proxy::lm_proxy_local.

8.35.3.2 virtual lm_slow_query_proxy& uva::smt::bpbd::server::lm::proxy::lm_proxy::allocate_slow_query_proxy ( ) [pure virtual]

This method allows to get a trie query executor for the given trie

Returns

the trie query proxy object

Implemented in uva::smt::bpbd::server::lm::proxy::lm_proxy_local.

8.35.3.3 virtual void uva::smt::bpbd::server::lm::proxy::lm_proxy::connect (const lm_parameters & *params*) [pure virtual]

Allows to connect to the trie object based on the given parameters

Parameters

params	the model parameters

Implemented in uva::smt::bpbd::server::lm::proxy::lm_proxy_local.

8.35.3.4 virtual void uva::smt::bpbd::server::lm::proxy::lm_proxy::disconnect ( ) [pure virtual]

Allows to disconnect from the trie

Implemented in uva::smt::bpbd::server::lm::proxy::lm_proxy_local.

8.35.3.5 virtual void uva::smt::bpbd::server::lm::proxy::lm_proxy::dispose_fast_query_proxy (lm_fast_query_proxy & *query*) [pure virtual]

Dispose the previously allocated trie query object

Parameters

query	the trie query to dispose

Implemented in uva::smt::bpbd::server::lm::proxy::lm_proxy_local.

8.35.3.6 virtual void uva::smt::bpbd::server::lm::proxy::lm_proxy::dispose_slow_query_proxy (lm_slow_query_proxy & *query*) [pure virtual]

Dispose the previously allocated trie query object

Parameters

query	the trie query to dispose

Implemented in uva::smt::bpbd::server::lm::proxy::lm_proxy_local.

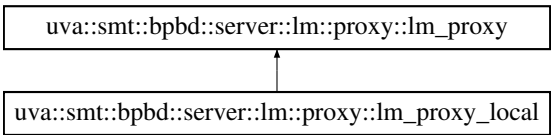
The documentation for this class was generated from the following file:

- inc/server/lm/proxy/lm_proxy.hpp

8.36 uva::smt::bpbd::server::lm::proxy::lm_proxy_local Class Reference

`#include <lm_proxy_local.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::proxy::lm_proxy_local:



Public Member Functions

- lm_proxy_local ()
- virtual ~lm_proxy_local ()
- virtual void connect (const lm_parameters &params)
- virtual void disconnect ()
- virtual lm_fast_query_proxy & allocate_fast_query_proxy ()

- virtual void dispose_fast_query_proxy (lm_fast_query_proxy &query)
- virtual lm_slow_query_proxy & allocate_slow_query_proxy ()
- virtual void dispose_slow_query_proxy (lm_slow_query_proxy &query)

Protected Attributes

- lm_word_index m_word_index
- lm_model_type m_model
- prob_weight m_unk_word_prob
- word_uid m_begin_tag_uid
- word_uid m_end_tag_uid

8.36.1 Detailed Description

This is a local trie proxy implementation of the trie proxy interface. Here we do not connect to remote server or something but rather work with a locally loaded trie model.

Definition at line 67 of file lm_proxy_local.hpp.

8.36.2 Constructor & Destructor Documentation

8.36.2.1 uva::smt::bpbd::server::lm::proxy::lm_proxy_local::lm_proxy_local ( ) [inline]

The basic constructor of the trie proxy implementation class

Parameters

params	the language model parameters

Definition at line 74 of file lm_proxy_local.hpp.

8.36.2.2 virtual uva::smt::bpbd::server::lm::proxy::lm_proxy_local::~lm_proxy_local ( ) [inline], [virtual]

See also

lm_proxy

Definition at line 80 of file lm_proxy_local.hpp.

8.36.3 Member Function Documentation

8.36.3.1 virtual lm_fast_query_proxy& uva::smt::bpbd::server::lm::proxy::lm_proxy_local::allocate_fast_query_proxy ( ) [inline], [virtual]

Todo {In the future we should just use a number of stack allocated objects in order to reduce the new/delete overhead}

See also

lm_proxy

Implements uva::smt::bpbd::server::lm::proxy::lm_proxy.

Definition at line 114 of file lm_proxy_local.hpp.

8.36.3.2 virtual lm_slow_query_proxy& uva::smt::bpbd::server::lm::proxy::lm_proxy_local::allocate_slow_query_proxy ( ) [inline], [virtual]

Todo {In the future we should just use a number of stack allocated objects in order to reduce the new/delete overhead}

See also

lm_proxy

Implements uva::smt::bpbd::server::lm::proxy::lm_proxy.

Definition at line 132 of file lm_proxy_local.hpp.

8.36.3.3 virtual void uva::smt::bpbd::server::lm::proxy::lm_proxy_local::connect (const lm_parameters & *params*) [inline], [virtual]

See also

lm_proxy

Implements uva::smt::bpbd::server::lm::proxy::lm_proxy.

Definition at line 88 of file lm_proxy_local.hpp.

8.36.3.4 virtual void uva::smt::bpbd::server::lm::proxy::lm_proxy_local::disconnect ( ) [inline], [virtual]

See also

lm_proxy

Implements uva::smt::bpbd::server::lm::proxy::lm_proxy.

Definition at line 104 of file lm_proxy_local.hpp.

8.36.3.5 virtual void uva::smt::bpbd::server::lm::proxy::lm_proxy_local::dispose_fast_query_proxy (lm_fast_query_proxy & *query*) [inline], [virtual]

Todo {In the future we should just use a number of stack allocated objects in order to reduce the new/delete overhead}

See also

lm_proxy

Implements uva::smt::bpbd::server::lm::proxy::lm_proxy.

Definition at line 123 of file lm_proxy_local.hpp.

8.36.3.6 virtual void uva::smt::bpbd::server::lm::proxy::lm_proxy_local::dispose_slow_query_proxy (lm_slow_query_proxy & *query*) [inline], [virtual]

Todo {In the future we should just use a number of stack allocated objects in order to reduce the new/delete overhead}

See also

lm_proxy

Implements uva::smt::bpbd::server::lm::proxy::lm_proxy.

Definition at line 141 of file lm_proxy_local.hpp.

8.36.4 Member Data Documentation

8.36.4.1 word_uid uva::smt::bpbd::server::lm::proxy::lm_proxy_local::m_begin_tag_uid [protected]

Definition at line 243 of file lm_proxy_local.hpp.

8.36.4.2 word_uid uva::smt::bpbd::server::lm::proxy::lm_proxy_local::m_end_tag_uid [protected]

Definition at line 244 of file lm_proxy_local.hpp.

8.36.4.3 lm_model_type uva::smt::bpbd::server::lm::proxy::lm_proxy_local::m_model [protected]

Definition at line 237 of file lm_proxy_local.hpp.

8.36.4.4 prob_weight uva::smt::bpbd::server::lm::proxy::lm_proxy_local::m_unk_word_prob [protected]

Definition at line 240 of file lm_proxy_local.hpp.

8.36.4.5 lm_word_index uva::smt::bpbd::server::lm::proxy::lm_proxy_local::m_word_index [protected]

Definition at line 234 of file lm_proxy_local.hpp.

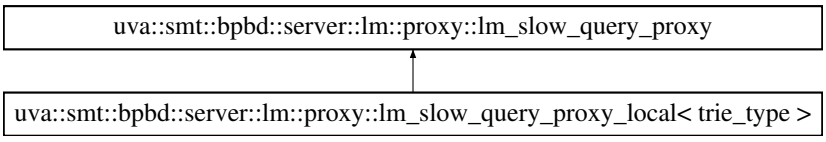
The documentation for this class was generated from the following file:

- inc/server/lm/proxy/lm_proxy_local.hpp

8.37 uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy Class Reference

`#include <lm_slow_query_proxy.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy:



Public Member Functions

- virtual ~lm_slow_query_proxy ()
- virtual void execute (text_piece_reader &line)=0

8.37.1 Detailed Description

This class represents a trie query proxy interface class. It allows to interact with templated trie queries in a uniform way.

Definition at line 48 of file lm_slow_query_proxy.hpp.

8.37.2 Constructor & Destructor Documentation

8.37.2.1 virtual uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy::~lm_slow_query_proxy ( ) [inline], [virtual]

The basic virtual destructor

Definition at line 54 of file lm_slow_query_proxy.hpp.

8.37.3 Member Function Documentation

8.37.3.1 virtual void uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy::execute (text_piece_reader & *line*) [pure virtual]

Allows to execute m-gram the query. The query starts with the m-gram size given by min_level and then grows until the maximum of LM_M_GRAM_LEVEL_MAX. After that m-grams of the LM_M_GRAM_LEVEL_MAX are computed via a sliding window: Let: "min_level == 2", "LM_MAX_QUERY_LEN = 4", "num_word_ids == 6" and "word_↩ ids == w1w2w3w4w5w6" Then this method will compute the sum: P(w2|w1) + P(w3|w1w2) + P(w4|w1w2w3) + P(w5|w2w3w4) + P(w6|w3w4w5)

Parameters

line	the text piece reader storing the m-gram query line

Implemented in uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local< trie_type >.

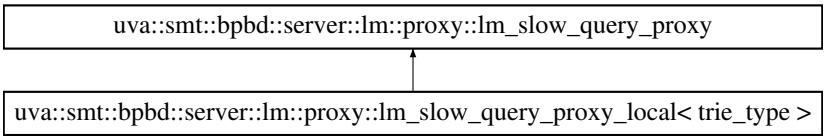
The documentation for this class was generated from the following file:

- inc/server/lm/proxy/lm_slow_query_proxy.hpp

8.38 uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local< trie_type > Class Template Reference

`#include <lm_slow_query_proxy_local.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local< trie_type >:



Public Types

- typedef trie_type::WordIndexType word_index_type

Public Member Functions

- lm_slow_query_proxy_local (const trie_type &trie)
- virtual ~lm_slow_query_proxy_local ()
- virtual void execute (text_piece_reader &line)

Protected Member Functions

- void get_report_interm_results (const phrase_length begin_word_idx, const phrase_length first_end_word↩ _idx, const phrase_length last_end_word_idx)
- void report_final_result ()
- string get_m_gram_str (const phrase_length begin_word_idx, const phrase_length end_word_idx) const
- string get_query_str () const
- virtual void set_tokens_and_word_ids (text_piece_reader phrase)

8.38.1 Detailed Description

template<typename trie_type>class uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local< trie_type >

This is a local implementation of the language model query This implementation works with the local trie

Definition at line 51 of file lm_slow_query_proxy_local.hpp.

8.38.2 Member Typedef Documentation

8.38.2.1 template<typename trie_type > typedef trie_type::WordIndexType uva::smt::bpbd::server::lm::proxy::lm_↩ slow_query_proxy_local< trie_type >::word_index_type

Definition at line 54 of file lm_slow_query_proxy_local.hpp.

8.38.3 Constructor & Destructor Documentation

8.38.3.1 template<typename trie_type > uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local< trie_type >::lm_slow_query_proxy_local (const trie_type & *trie*) [inline]

The basic constructor that accepts the trie reference to query to

Parameters

trie	the trie to query

Definition at line 60 of file lm_slow_query_proxy_local.hpp.

8.38.3.2 template<typename trie_type > virtual uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local< trie_type >::~lm_slow_query_proxy_local ( ) [inline], [virtual]

See also

lm_query_proxy

Definition at line 68 of file lm_slow_query_proxy_local.hpp.

8.38.4 Member Function Documentation

8.38.4.1 template<typename trie_type > virtual void uva::smt::bpbd::server::lm::proxy::lm_slow↩ _query_proxy_local< trie_type >::execute (text_piece_reader & *line*) [inline], [virtual]

Allows to execute m-gram the query. The query starts with the m-gram size given by min_level and then grows until the maximum of LM_M_GRAM_LEVEL_MAX. After that m-grams of the LM_M_GRAM_LEVEL_MAX are computed

via a sliding window: Let: "min_level == 2", "LM_MAX_QUERY_LEN = 4", "num_word_ids == 6" and "word_↩ ids == w1w2w3w4w5w6" Then this method will compute the sum: P(w2|w1) + P(w3|w1w2) + P(w4|w1w2w3) + P(w5|w2w3w4) + P(w6|w3w4w5)

Parameters

line	the text piece reader storing the m-gram query line

Implements uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy.

Definition at line 83 of file lm_slow_query_proxy_local.hpp.

8.38.4.2 template<typename trie_type > string uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local< trie_type >::get_m_gram_str (const phrase_length *begin_word_idx,* const phrase_length *end_word_idx*) const [inline], [protected]

For the given N-gram, for some level M <=N , this method allows to give the string of the object for which the probability is computed, e.g.: N-gram = "word1" -> result = "word1" N-gram = "word1 word2 word3" -> result = "word3 | word1 word2" for the first M tokens of the N-gram

Parameters

begin_word_idx	the m-gram's begin word index
end_word_idx	the m-gram's begin word index

Returns

the resulting string

Definition at line 191 of file lm_slow_query_proxy_local.hpp.

8.38.4.3 template<typename trie_type > string uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local< trie_type >::get_query_str ( ) const [inline], [protected]

For the given N-gram, this method allows to give the string of the object for which the probability is computed, e.g.: N-gram = "word1" -> result = "word1" N-gram = "word1 word2 word3" -> result = "word1 word2 word3"

Returns

the resulting string

Definition at line 215 of file lm_slow_query_proxy_local.hpp.

8.38.4.4 template<typename trie_type > void uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local< trie_type >::get_report_interm_results (const phrase_length *begin_word_idx,* const phrase_length *first_end_word_idx,* const phrase_length *last_end_word_idx*) [inline], [protected]

Allows report the intermediate results of the loose sub-sub queries defined by the arguments

Parameters

begin_word_idx	the sub query begin word index
first_end_word_idx	the first sub-sub query end word index
last_end_word_idx	the last sub-sub query end word index

Definition at line 148 of file lm_slow_query_proxy_local.hpp.

8.38.4.5 template<typename trie_type > void uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local< trie_type >::report_final_result ( ) [inline], [protected]

Allows to report the total joint probability of the query

Definition at line 170 of file lm_slow_query_proxy_local.hpp.

8.38.4.6 template<typename trie_type > virtual void uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local< trie_type >::set_tokens_and_word_ids (text_piece_reader *phrase*) [inline], [protected], [virtual]

Allows to parse the m-gram into the tokens and get the word ids

Definition at line 235 of file lm_slow_query_proxy_local.hpp.

The documentation for this class was generated from the following file:

- inc/server/lm/proxy/lm_slow_query_proxy_local.hpp

8.39 uva::utils::logging::logger Class Reference

`#include <logger.hpp>`

Public Member Functions

- virtual ~logger ()

Static Public Member Functions

- static void get_reporting_levels (vector< string > *p_reporting_levels)
- static void set_reporting_level (const string level)
- static std::ostream & get (debug_levels_enum level)
- static std::ostream & get (debug_levels_enum level, const char *file, const char *func, const char *line)
- static bool is_relevant_level (const debug_levels_enum &level)
- static debug_levels_enum & get_reporting_level ()
- static const string get_curr_level_str ()
- static void start_progress_bar (const string &msg)
- static void update_progress_bar ()
- static void stop_progress_bar ()
- static bool is_progress_bar_on ()

8.39.1 Detailed Description

This is a trivial logging facility that exchibits a singleton behavior and does output to stderr and stdout.

Definition at line 147 of file logger.hpp.

8.39.2 Constructor & Destructor Documentation

8.39.2.1 virtual uva::utils::logging::logger::~logger ( ) [inline], [virtual]

Definition at line 150 of file logger.hpp.

8.39.3 Member Function Documentation

8.39.3.1 static std::ostream& uva::utils::logging::logger::get (debug_levels_enum *level*) [inline], [static]

This methods allows to get the output stream for the given log-level

Parameters

level	the log level for the messages to print

Returns

the output stream object

Definition at line 170 of file logger.hpp.

8.39.3.2 static std::ostream& uva::utils::logging::logger::get (debug_levels_enum *level*, const char * *file*, const char * *func*, const char * *line*) [inline], [static]

This methods allows to get the output stream for the given log-level

Parameters

level	the log level for the messages to print

Returns

the output stream object

Definition at line 179 of file logger.hpp.

8.39.3.3 static const string uva::utils::logging::logger::get_curr_level_str ( ) [inline], [static]

Allows to obtain the current reporting level string

Returns

the current reporting level string

Definition at line 205 of file logger.hpp.

8.39.3.4 static debug_levels_enum& uva::utils::logging::logger::get_reporting_level ( ) [inline], [static]

Returns the reference to the internal log level variable

Returns

the reference to the internal log level variable

Definition at line 197 of file logger.hpp.

8.39.3.5 void uva::utils::logging::logger::get_reporting_levels (vector< string > * *p_reporting_levels*) [static]

Allows to retrieve the list of supporter logging levels

Parameters

p_reporting_levels	the pointer to the logging levels vector to be filled in

Definition at line 86 of file logger.cpp.

8.39.3.6 static bool uva::utils::logging::logger::is_progress_bar_on ( ) [inline], [static]

The function allows to check if the progress bar is running or not

Returns

true if the progress bar is running, otherwise case;

Definition at line 232 of file logger.hpp.

8.39.3.7 static bool uva::utils::logging::logger::is_relevant_level (const debug_levels_enum & *level*) [inline], [static]

Checks if the current reporting level is higher or equal to the given

Returns

the reporting level to check
true if the given reporting level is smaller or equal to the current, otherwise false

Definition at line 189 of file logger.hpp.

8.39.3.8 void uva::utils::logging::logger::set_reporting_level (const string *level*) [static]

Allows to set the logging level from a string, if not recognized - reports a warning!

Parameters

level	the string level to set

Todo {This function is ugly improve it by using a map, or a similar so that we could just get an appropriate level for the string.}

Definition at line 99 of file logger.cpp.

8.39.3.9 void uva::utils::logging::logger::start_progress_bar (const string & *msg*) [static]

The function that start progress bar Works if the current debug level is <= INFO

Parameters

msg	the message to display

Definition at line 186 of file logger.cpp.

8.39.3.10 void uva::utils::logging::logger::stop_progress_bar ( ) [static]

The function that stops progress bar Works if the current debug level is <= INFO

Definition at line 227 of file logger.cpp.

8.39.3.11 void uva::utils::logging::logger::update_progress_bar ( ) [static]

The function that updates progress bar Works if the current debug level is <= INFO

Definition at line 210 of file logger.cpp.

The documentation for this class was generated from the following files:

- inc/common/utils/logging/logger.hpp
- src/common/utils/logging/logger.cpp

8.40 uva::utils::logging::logging_synch Struct Reference

`#include <logger.hpp>`

Public Types

- typedef lock_guard< recursive_mutex > rec_scoped_lock

Static Public Attributes

- static recursive_mutex mv

8.40.1 Detailed Description

This structures stores the recursive synchronization mutex for logging. The mutex is to be recursive as functions called when logging can do the own logging.

Definition at line 65 of file logger.hpp.

8.40.2 Member Typedef Documentation

8.40.2.1 typedef lock_guard<recursive_mutex> uva::utils::logging::logging_synch::rec_scoped_lock

Definition at line 67 of file logger.hpp.

8.40.3 Member Data Documentation

8.40.3.1 recursive_mutex uva::utils::logging::logging_synch::mv [static]

Definition at line 69 of file logger.hpp.

The documentation for this struct was generated from the following files:

- inc/common/utils/logging/logger.hpp
- src/common/utils/logging/logger.cpp

8.41 uva::smt::bpbd::server::lm::m_grams::m_gram_payload_s Struct Reference

`#include <m_gram_payload.hpp>`

Public Member Functions

- m_gram_payload_s ()
- m_gram_payload_s (prob_weight prob, prob_weight back)

Public Attributes

- prob_weight m_prob
- prob_weight m_back

8.41.1 Detailed Description

This data structure stores the probability and back off weight payload for an m-gram

Definition at line 63 of file m_gram_payload.hpp.

8.41.2 Constructor & Destructor Documentation

8.41.2.1 uva::smt::bpbd::server::lm::m_grams::m_gram_payload_s::m_gram_payload_s ( ) [inline], [explicit]

Definition at line 67 of file m_gram_payload.hpp.

8.41.2.2 uva::smt::bpbd::server::lm::m_grams::m_gram_payload_s::m_gram_payload_s (prob_weight *prob,* prob_weight *back*) [inline]

Definition at line 72 of file m_gram_payload.hpp.

8.41.3 Member Data Documentation

8.41.3.1 prob_weight uva::smt::bpbd::server::lm::m_grams::m_gram_payload_s::m_back

Definition at line 65 of file m_gram_payload.hpp.

8.41.3.2 prob_weight uva::smt::bpbd::server::lm::m_grams::m_gram_payload_s::m_prob

Definition at line 64 of file m_gram_payload.hpp.

The documentation for this struct was generated from the following file:

- inc/server/lm/mgrams/m_gram_payload.hpp

8.42 uva::smt::bpbd::server::lm::m_gram_query Class Reference

`#include <m_gram_query.hpp>`

Public Types

- typedef const void * payload_ptr

Public Member Functions

- m_gram_query ()
- template<bool is_need_ctx_ids>
  void set_data (const phrase_length num_words, const word_uid *word_ids)
- phrase_length get_query_begin_word_idx () const
- phrase_length get_query_end_word_idx () const

- void set_word_indxes (const phrase_length sub_query_begin_word_idx, const phrase_length sub_sub_↩ query_first_end_word_idx, const phrase_length sub_query_end_word_idx)
- void set_word_indxes (const phrase_length sub_query_begin_word_idx, const phrase_length sub_query_↩ end_word_idx)
- word_uid operator[ ] (const phrase_length idx) const
- bool is_not_finished () const
- uint64_t get_curr_m_gram_hash ()
- word_uid get_curr_begin_word_id () const
- word_uid get_curr_end_word_id () const
- word_uid get_curr_uni_gram_word_id () const
- void set_curr_payload (const void ∗payload)
- const void ∗& get_curr_payload_ref ()
- bool is_curr_uni_gram () const
- phrase_length get_curr_level () const
- phrase_length get_curr_level_m1 () const
- phrase_length get_curr_level_m2 () const
- const TM_Gram_Id_Value_Ptr get_curr_m_gram_id (uint8_t &len_bytes)
- TLongId & get_curr_ctx_ref ()

Public Attributes

- prob_weight m_probs [QUERY_M_GRAM_MAX_LEN]
- phrase_length m_curr_begin_word_idx
- phrase_length m_curr_end_word_idx

Friends

- ostream & operator<< (ostream &stream, const m_gram_query &value)

8.42.1 Detailed Description

This structure stores the basic data required for a query execution.

Parameters

m_query	the m-gram query itself
m_payloads	the two dimensional array of the payloads
m_last_ctx_ids	stores the last context id computed for the given row of the sub-m-gram matrix
m_probs	the array f probabilities
m_begin_word_↩ _idx	the currently considered begin word index
m_end_word_↩ idx	the currently considered end word index

Definition at line 65 of file m_gram_query.hpp.

8.42.2 Member Typedef Documentation

8.42.2.1 typedef const void∗ uva::smt::bpbd::server::lm::m_gram_query::payload_ptr

Definition at line 68 of file m_gram_query.hpp.

8.42.3 Constructor & Destructor Documentation

8.42.3.1 uva::smt::bpbd::server::lm::m_gram_query::m_gram_query ( ) [inline]

The basic constructor that gets a reference to the word index

Parameters

word_index	the reference to the word index

Definition at line 84 of file m_gram_query.hpp.

8.42.4 Member Function Documentation

8.42.4.1 word_uid uva::smt::bpbd::server::lm::m_gram_query::get_curr_begin_word_id ( ) const [inline]

Allows to get the current begin word id

Returns

the current begin word id

Definition at line 195 of file m_gram_query.hpp.

8.42.4.2 TLongId& uva::smt::bpbd::server::lm::m_gram_query::get_curr_ctx_ref ( ) [inline]

Allows to get a reference to the current context

Returns

the reference to the variable storing the current context value

Definition at line 289 of file m_gram_query.hpp.

8.42.4.3 word_uid uva::smt::bpbd::server::lm::m_gram_query::get_curr_end_word_id ( ) const [inline]

Allows to get the current end word id

Returns

the current end word id

Definition at line 203 of file m_gram_query.hpp.

8.42.4.4 phrase_length uva::smt::bpbd::server::lm::m_gram_query::get_curr_level ( ) const [inline]

Allows to get the level of the currently considered m-gram

Returns

the level of the currently considered m-gram

Definition at line 253 of file m_gram_query.hpp.

8.42.4.5 phrase_length uva::smt::bpbd::server::lm::m_gram_query::get_curr_level_m1 ( ) const [inline]

Allows to get the "level - 1" of the currently considered m-gram

Returns

the "level - 1" of the currently considered m-gram

Definition at line 261 of file m_gram_query.hpp.

8.42.4.6 phrase_length uva::smt::bpbd::server::lm::m_gram_query::get_curr_level_m2 ( ) const [inline]

Allows to get the "level - 2" of the currently considered m-gram

Returns

the "level - 2" of the currently considered m-gram

Definition at line 269 of file m_gram_query.hpp.

8.42.4.7 uint64_t uva::smt::bpbd::server::lm::m_gram_query::get_curr_m_gram_hash ( ) [inline]

Allows to compute the hash value of the m-gram defined by the current begin and end word indexes

Returns

the hash of the current m-gram

Definition at line 187 of file m_gram_query.hpp.

8.42.4.8 const TM_Gram_Id_Value_Ptr uva::smt::bpbd::server::lm::m_gram_query::get_curr_m_gram_id (uint8_t & *len_bytes*) [inline]

Allows to create a new m-gram id for the current m-gram defined by the current begin and end word index values. For the argument reference to the id data pointer the following holds: a) If there was no memory allocated for the M-gram id then there will be allocated as much as needed to store the given id. b) If there was memory allocated then no re-allocation will be done, then it is assumed that enough memory was allocated

Parameters

p_m_gram_id	the reference to the M-gram id data pointer to be initialized with the M-gram id data, must be pre-allocated

Definition at line 281 of file m_gram_query.hpp.

8.42.4.9 const void* & uva::smt::bpbd::server::lm::m_gram_query::get_curr_payload_ref ( ) [inline]

Allows to set the payload of the current m-gram defined by the current begin and end word indexes

Parameters

payload	the payload to be set

Definition at line 237 of file m_gram_query.hpp.

8.42.4.10 word_uid uva::smt::bpbd::server::lm::m_gram_query::get_curr_uni_gram_word_id ( ) const [inline]

Allows to get the word if of the current uni-gram This method shall only be called in case: m_curr_begin_word_idx == m_curr_end_word_idx

Returns

the word id of the current uni-gram

Definition at line 213 of file m_gram_query.hpp.

8.42.4.11 phrase_length uva::smt::bpbd::server::lm::m_gram_query::get_query_begin_word_idx ( ) const [inline]

Allows to get the begin word index of the query

Returns

the begin word index of the query

Definition at line 118 of file m_gram_query.hpp.

8.42.4.12 phrase_length uva::smt::bpbd::server::lm::m_gram_query::get_query_end_word_idx ( ) const [inline]

Allows to get the end word index of the query

Returns

the end word index of the query

Definition at line 126 of file m_gram_query.hpp.

8.42.4.13 bool uva::smt::bpbd::server::lm::m_gram_query::is_curr_uni_gram ( ) const [inline]

Allows to check if the current m-gram is a uni-gram

Returns

true if the current m-gram is a uni-gram, otherwise false

Definition at line 245 of file m_gram_query.hpp.

8.42.4.14 bool uva::smt::bpbd::server::lm::m_gram_query::is_not_finished ( ) const [inline]

Allows to check if the current sub-query execution is over or not

Returns

true if the sub-query execution is not finished yet

Definition at line 177 of file m_gram_query.hpp.

8.42.4.15 word_uid uva::smt::bpbd::server::lm::m_gram_query::operator[ ] (const phrase_length *idx*) const [inline]

Allows to retrieve the word id under the given index

Parameters

idx	the index of the word we need an id for

Returns

the word id

Definition at line 168 of file m_gram_query.hpp.

8.42.4.16 void uva::smt::bpbd::server::lm::m_gram_query::set_curr_payload (const void * *payload*) [inline]

Allows to set the payload of the current m-gram defined by the current begin and end word indexes

Parameters

payload	the payload to be set

Definition at line 228 of file m_gram_query.hpp.

8.42.4.17 template<bool is_need_ctx_ids> void uva::smt::bpbd::server::lm::m_gram_query::set_data (const phrase_length *num_words,* const word_uid * *word_ids*) [inline]

Allows to set new data to the query

Parameters

num_word_ids	stores the number of word ids, the maximum number of words must be QUERY_LENGTH
word_ids	the word identifiers of the words of the target phrase to compute the probability for

Definition at line 95 of file m_gram_query.hpp.

8.42.4.18 void uva::smt::bpbd::server::lm::m_gram_query::set_word_indxes (const phrase_length *sub_query_begin_word_idx,* const phrase_length *sub_sub_query_first_end_word_idx,* const phrase_length *sub_query_end_word_idx*) [inline]

Allows to set the begin and end m-gram word index. These define the m-gram for which the probability is to be computed. This method is handy for when we need streaming for a number of sub-sub m-grams starting in the same word but of the incremented length

Parameters

sub_query_↩ begin_word_idx	the sub-query begin word index
first_sub_sub_↩ query_end_↩ word_idx	the sub-sub query first end word index
sub_query_↩ end_word_idx	the sub query end word index

Definition at line 139 of file m_gram_query.hpp.

8.42.4.19 void uva::smt::bpbd::server::lm::m_gram_query::set_word_indxes (const phrase_length *sub_query_begin_word_idx,* const phrase_length *sub_query_end_word_idx*) [inline]

Allows to set the begin and end m-gram word index. These define the m-gram for which the probability is to be computed. This method is needed for when we only need one m-gram probability without streaming

Parameters

sub_query_↩ begin_word_idx	the sub-query begin word index
sub_query_↩ end_word_idx	the sub query end word index

Definition at line 155 of file m_gram_query.hpp.

8.42.5 Friends And Related Function Documentation

8.42.5.1 ostream& operator<< (ostream & *stream,* const m_gram_query & *value*) [friend]

Allows to serialize the m-gram query to the output stream as a string

Parameters

stream	the reference to the stream to output into
query	the query object to output

Returns

the reference to the stream

Definition at line 40 of file m_gram_query.cpp.

8.42.6 Member Data Documentation

8.42.6.1 phrase_length uva::smt::bpbd::server::lm::m_gram_query::m_curr_begin_word_idx

Definition at line 76 of file m_gram_query.hpp.

8.42.6.2 phrase_length uva::smt::bpbd::server::lm::m_gram_query::m_curr_end_word_idx

Definition at line 78 of file m_gram_query.hpp.

8.42.6.3 prob_weight uva::smt::bpbd::server::lm::m_gram_query::m_probs[QUERY_M_GRAM_MAX_LEN]

Definition at line 73 of file m_gram_query.hpp.

The documentation for this class was generated from the following file:

- inc/server/lm/models/m_gram_query.hpp

8.43 uva::utils::containers::mem_increase_strategy Class Reference

`#include <dynamic_memory_arrays.hpp>`

Public Member Functions

- mem_increase_strategy (const mem_inc_types_enum &stype, const TCapacityIncFunct get_capacity_inc↩ _func, const size_t min_mem_inc, const size_t mem_inc_factor)
- mem_increase_strategy ()
- mem_increase_strategy (const mem_increase_strategy &other)
- string get_strategy_info () const
- const size_t get_new_capacity (const size_t capacity) const

8.43.1 Detailed Description

This class stores the memory increment strategy and allows to use it

Definition at line 74 of file dynamic_memory_arrays.hpp.

8.43.2 Constructor & Destructor Documentation

8.43.2.1 uva::utils::containers::mem_increase_strategy::mem_increase_strategy (const mem_inc_types_enum & *stype,* const TCapacityIncFunct *get_capacity_inc_func,* const size_t *min_mem_inc,* const size_t *mem_inc_factor*) [inline]

The main constructor to be used

Parameters

stype	the strategy type
get_capacity_↩ inc_func	the strategy function
min_mem_inc	the minimum memory increase in number of elements
mem_inc_factor	the memory increment factor, the number we will multiply by the computed increment

Definition at line 94 of file dynamic_memory_arrays.hpp.

8.43.2.2 uva::utils::containers::mem_increase_strategy::mem_increase_strategy ( ) [inline]

Definition at line 104 of file dynamic_memory_arrays.hpp.

8.43.2.3 uva::utils::containers::mem_increase_strategy::mem_increase_strategy (const mem_increase_strategy & *other*) [inline]

Definition at line 109 of file dynamic_memory_arrays.hpp.

8.43.3 Member Function Documentation

8.43.3.1 const size_t uva::utils::containers::mem_increase_strategy::get_new_capacity (const size_t *capacity*) const [inline]

Compute the new capacity given the provided one, this function used the capacity increase function stored in m_↩ get_capacity_inc_func.

Parameters

capacity	the current capacity

Returns

the proposed capacity increase

Definition at line 132 of file dynamic_memory_arrays.hpp.

8.43.3.2 string uva::utils::containers::mem_increase_strategy::get_strategy_info ( ) const [inline]

Allows to retrieve the strategy name

Returns

the strategy name

Definition at line 118 of file dynamic_memory_arrays.hpp.

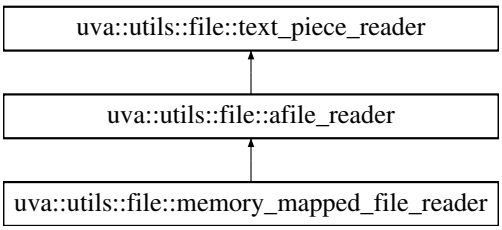
The documentation for this class was generated from the following file:

- inc/common/utils/containers/dynamic_memory_arrays.hpp

8.44 uva::utils::file::memory_mapped_file_reader Class Reference

`#include <memory_mapped_file_reader.hpp>`

Inheritance diagram for uva::utils::file::memory_mapped_file_reader:



Public Member Functions

- memory_mapped_file_reader (const char *fileName)
- virtual void log_reader_type_info ()
- bool get_first_line (text_piece_reader &out)
- virtual bool is_open () const
- virtual operator bool () const
- virtual void close ()

Additional Inherited Members

8.44.1 Detailed Description

This is the file reader for the memory mapped file. It is supposed to provide fast memory reads from large files.

For more information on memory mapped files read: https://en.wikipedia.org/wiki/↩ Memory-mapped_file

A possible benefit of memory-mapped files is a "lazy loading", thus using small amounts of RAM even for a very large file. Trying to load the entire contents of a file that is significantly larger than the amount of memory available can cause severe thrashing as the operating system reads from disk into memory and simultaneously writes pages from memory back to disk. Memory-mapping may not only bypass the page file completely, but the system only needs to load the smaller page-sized sections as data is being edited, similarly to demand paging scheme used for programs.

The memory mapping process is handled by the virtual memory manager, which is the same subsystem responsible for dealing with the page file. Memory mapped files are loaded into memory one entire page at a time. The page size is selected by the operating system for maximum performance. Since page file management is one of the most critical elements of a virtual memory system, loading page sized sections of a file into physical memory is typically a very highly optimized system function.

Here is also some nice explanation from: http://stackoverflow.com/questions/1972765/mmap-problem-alloc

Mapping the file into memory is different to actually reading the file into memory. Were you to read it in, you would have to transfer the entire contents into memory. By mapping it, you let the operating system handle it. If you attempt to read or write to a location in that memory area, the OS will load the relevant section for you first. It will not load the entire file unless the entire file is needed.

That is where you get your performance gain. If you map the entire file but only change one byte then unmap it, you'll find that there's not much disk I/O at all.

Of course, if you touch every byte in the file, then yes, it will all be loaded at some point but not necessarily in physical RAM all at once. But that's the case even if you load the entire file up front. The OS will swap out parts of your data if there's not enough physical memory to contain it all, along with that of the other processes in the system.

The main advantages of memory mapping are:

1) You defer reading the file sections until they're needed (and, if they're never needed, they don't get loaded). So there's no big upfront cost as you load the entire file. It amortises the cost of loading.

2) The writes are automated, you don't have to write out every byte. Just close it and the OS will write out the changed sections. I think this also happens when the memory is swapped out as well (in low physical memory situations), since your buffer is simply a window onto the file.

Definition at line 103 of file memory_mapped_file_reader.hpp.

8.44.2 Constructor & Destructor Documentation

8.44.2.1 uva::utils::file::memory_mapped_file_reader::memory_mapped_file_reader (const char * *fileName*) [inline]

The basic constructor

Parameters

fileName	the file name

Definition at line 115 of file memory_mapped_file_reader.hpp.

8.44.3 Member Function Documentation

8.44.3.1 virtual void uva::utils::file::memory_mapped_file_reader::close ( ) [inline], [virtual]

This method should be used to close the file

Reimplemented from uva::utils::file::afile_reader.

Definition at line 183 of file memory_mapped_file_reader.hpp.

8.44.3.2 bool uva::utils::file::memory_mapped_file_reader::get_first_line (text_piece_reader & *out*) [inline]

Definition at line 160 of file memory_mapped_file_reader.hpp.

8.44.3.3 virtual bool uva::utils::file::memory_mapped_file_reader::is_open ( ) const [inline], [virtual]

This method is used to check if the file was successfully opened.

Returns

true if the file is successfully opened otherwise false.

Implements uva::utils::file::afile_reader.

Definition at line 168 of file memory_mapped_file_reader.hpp.

8.44.3.4 virtual void uva::utils::file::memory_mapped_file_reader::log_reader_type_info ( ) [inline], [virtual]

Allows to log the information about the instantiated file reader type

Implements uva::utils::file::afile_reader.

Definition at line 156 of file memory_mapped_file_reader.hpp.

8.44.3.5 virtual uva::utils::file::memory_mapped_file_reader::operator bool ( ) const [inline], [explicit], [virtual]

Checks if the file is present.

Returns

true if it is

Implements uva::utils::file::afile_reader.

Definition at line 176 of file memory_mapped_file_reader.hpp.

The documentation for this class was generated from the following file:

- inc/common/utils/file/memory_mapped_file_reader.hpp

8.45 uva::utils::monitor::memory_usage Struct Reference

`#include <statistics_monitor.hpp>`

Public Member Functions

- memory_usage ()

Public Attributes

- int vmsize
- int vmpeak
- int vmrss
- int vmhwm

8.45.1 Detailed Description

This structure stores the memory statistics. Resident Set Size: number of pages the process has in real memory. This is just the pages which count toward text, data, or stack space. This does not include pages which have not been demand-loaded in, or which are swapped out. For more information see http://man7.org/linux/man-pages/man5/proc.5.html

Definition at line 46 of file statistics_monitor.hpp.

8.45.2 Constructor & Destructor Documentation

8.45.2.1 uva::utils::monitor::memory_usage::memory_usage ( ) [inline]

Definition at line 56 of file statistics_monitor.hpp.

8.45.3 Member Data Documentation

8.45.3.1 int uva::utils::monitor::memory_usage::vmhwm

Definition at line 54 of file statistics_monitor.hpp.

8.45.3.2 int uva::utils::monitor::memory_usage::vmpeak

Definition at line 50 of file statistics_monitor.hpp.

8.45.3.3 int uva::utils::monitor::memory_usage::vmrss

Definition at line 52 of file statistics_monitor.hpp.

8.45.3.4 int uva::utils::monitor::memory_usage::vmsize

Definition at line 48 of file statistics_monitor.hpp.

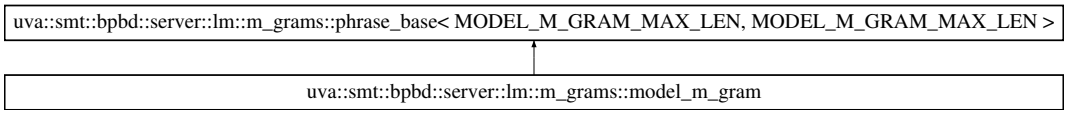
The documentation for this struct was generated from the following file:

- inc/common/utils/monitor/statistics_monitor.hpp

8.46 uva::smt::bpbd::server::lm::m_grams::model_m_gram Class Reference

`#include <model_m_gram.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::m_grams::model_m_gram:



Public Types

- typedef phrase_base< MODEL_M_GRAM_MAX_LEN, MODEL_M_GRAM_MAX_LEN > BASE

Public Member Functions

- model_m_gram (phrase_length actual_level)
- void start_new_m_gram ()
- text_piece_reader & get_next_new_token ()
- bool is_unk_unigram () const
- template<typename WordIndexType >
  void prepare_for_adding (WordIndexType &word_index)
- uint64_t get_hash () const

Public Attributes

- m_gram_payload m_payload
- prob_weight m_prob
- prob_weight m_back_off

Friends

- ostream & operator<< (ostream &stream, const model_m_gram &gram)

Additional Inherited Members

8.46.1 Detailed Description

This class is used to represent the N-Gram that will be stored into the language model.

Definition at line 57 of file model_m_gram.hpp.

8.46.2 Member Typedef Documentation

8.46.2.1 typedef phrase_base<MODEL_M_GRAM_MAX_LEN, MODEL_M_GRAM_MAX_LEN> uva::smt::bpbd::server::lm::m_grams::model_m_gram::BASE

Definition at line 60 of file model_m_gram.hpp.

8.46.3 Constructor & Destructor Documentation

8.46.3.1 uva::smt::bpbd::server::lm::m_grams::model_m_gram::model_m_gram (phrase_length *actual_level*) [inline]

The basic constructor, is to be used when the M-gram level is known beforehand. Allows to set the actual M-gram level to a concrete value.

Parameters

actual_level	the actual level of the m-gram that will be used should be <= MODEL_M_GRAM_MAX_LEN

Definition at line 77 of file model_m_gram.hpp.

8.46.4 Member Function Documentation

8.46.4.1 uint64_t uva::smt::bpbd::server::lm::m_grams::model_m_gram::get_hash ( ) const [inline]

Allows to retrieve the hash value for the given m-gram

Returns

the hash value for the given m-gram

Definition at line 163 of file model_m_gram.hpp.

8.46.4.2 text_piece_reader& uva::smt::bpbd::server::lm::m_grams::model_m_gram::get_next_new_token ( ) [inline]

Returns the reference to the next new token of the m-gram

Returns

the reference to the next new token of the m-gram

Definition at line 93 of file model_m_gram.hpp.

8.46.4.3 bool uva::smt::bpbd::server::lm::m_grams::model_m_gram::is_unk_unigram ( ) const [inline]

Allows to detect whether the given m-gram is an <unk> unigram.

Returns

true if this is an <unk> unigram

Definition at line 105 of file model_m_gram.hpp.

8.46.4.4 template<typename WordIndexType > void uva::smt::bpbd::server::lm::m_grams::model_m_gram::prepare_for_adding (WordIndexType & *word_index*) [inline]

Allows to prepare the M-gram for being used for adding it to the trie This includes registering the one gram in the word index

Parameters

word_index	the word index to be used

Definition at line 116 of file model_m_gram.hpp.

8.46.4.5 void uva::smt::bpbd::server::lm::m_grams::model_m_gram::start_new_m_gram ( ) [inline]

Allows to start a new M-gram with the given level

Parameters

CURR_LEVEL	the level of the M-gram we are starting

Definition at line 85 of file model_m_gram.hpp.

8.46.5 Friends And Related Function Documentation

8.46.5.1 ostream& operator<< (ostream & *stream,* const model_m_gram & *gram*) [friend]

Allows to serialize the m-gram to the output stream as a string

Parameters

stream	the reference to the stream to output into
gram	the m-gram object to output

Returns

the reference to the stream

Definition at line 44 of file model_m_gram.cpp.

8.46.6 Member Data Documentation

8.46.6.1 prob_weight uva::smt::bpbd::server::lm::m_grams::model_m_gram::m_back_off

Definition at line 69 of file model_m_gram.hpp.

8.46.6.2 m_gram_payload uva::smt::bpbd::server::lm::m_grams::model_m_gram::m_payload

Definition at line 63 of file model_m_gram.hpp.

8.46.6.3 prob_weight uva::smt::bpbd::server::lm::m_grams::model_m_gram::m_prob

Definition at line 66 of file model_m_gram.hpp.

The documentation for this class was generated from the following file:

- inc/server/lm/mgrams/model_m_gram.hpp

8.47 uva::smt::bpbd::server::decoder::stack::multi_stack Class Reference

`#include <multi_stack.hpp>`

Public Member Functions

- multi_stack (const de_parameters &params, acr_bool_flag is_stop, const string &source_sent, const sentence_data_map &sent_data, const rm_query_proxy &rm_query, lm_fast_query_proxy &lm_query)
- ~multi_stack ()
- void expand ()
- void get_best_trans (string &target_sent) const

Protected Member Functions

- void add_stack_state (stack_state_ptr new_state)

8.47.1 Detailed Description

This is the translation stack class that is responsible for the sentence translation

Definition at line 73 of file multi_stack.hpp.

8.47.2 Constructor & Destructor Documentation

8.47.2.1 uva::smt::bpbd::server::decoder::stack::multi_stack::multi_stack (const de_parameters & *params,* acr_bool_flag *is_stop,* const string & *source_sent,* const sentence_data_map & *sent_data,* const rm_query_proxy & *rm_query,* lm_fast_query_proxy & *lm_query*) [inline]

The basic constructor

Parameters

params	the decoder parameters, stores the reference to it
is_stop	the stop flag
source_sent	the reference to the source sentence
sent_data	the retrieved sentence data
rm_query	the reordering model query
lm_query	the language model query object

Definition at line 85 of file multi_stack.hpp.

8.47.2.2 uva::smt::bpbd::server::decoder::stack::multi_stack::~multi_stack ( ) [inline]

The basic destructor

Definition at line 121 of file multi_stack.hpp.

8.47.3 Member Function Documentation

8.47.3.1 void uva::smt::bpbd::server::decoder::stack::multi_stack::add_stack_state (stack_state_ptr *new_state*) [inline], [protected]

Allows to add a new stack state into the proper stack level

Parameters

new_state	the new stack state, not NULL

Definition at line 191 of file multi_stack.hpp.

8.47.3.2 void uva::smt::bpbd::server::decoder::stack::multi_stack::expand ( ) [inline]

Allows to extend the hypothesis, when extending the stack we immediately re-combine

Definition at line 141 of file multi_stack.hpp.

8.47.3.3 void uva::smt::bpbd::server::decoder::stack::multi_stack::get_best_trans (string & *target_sent*) const [inline]

Allows to get the best translation from the stack after the decoding has finished.

Parameters

target_sent	[out] the variable to store the translation

Definition at line 172 of file multi_stack.hpp.

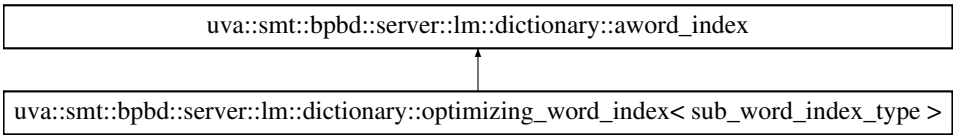
The documentation for this class was generated from the following file:

- inc/server/decoder/stack/multi_stack.hpp

8.48 uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type > Class Template Reference

`#include <optimizing_word_index.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type >:



Public Member Functions

- optimizing_word_index (const float memory_factor)
- void reserve (const size_t num_words)
- size_t get_number_of_words (const size_t num_words) const
- word_uid get_word_id (const text_piece_reader &token) const
- bool is_word_registering_needed () const
- word_uid register_word (const text_piece_reader &token)
- bool is_word_counts_needed () const
- void count_word (const text_piece_reader &word, prob_weight prob)

- void do_post_word_count ()
- bool is_post_actions_needed () const
- void do_post_actions ()
- virtual ~optimizing_word_index ()

Static Public Member Functions

- static constexpr bool is_word_index_continuous ()

8.48.1 Detailed Description

template<typename sub_word_index_type>class uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type >

This class is to be used as an optimizer wrapped around the original index. The main idea is that a word index is provided to this class and is used for initial data gathering. After that is done, during the post actions the data from the original word index is taken and converted into optimized format. This data is then stored within this class. The original word index is then destroyed to save space.

Parameters

sub_word_index_type	the sub WordIndex type to be used

Definition at line 85 of file optimizing_word_index.hpp.

8.48.2 Constructor & Destructor Documentation

8.48.2.1 template<typename sub_word_index_type > uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type >::optimizing_word_index (const float *memory_factor*) [inline]

This is the main constructor to be used. It accepts ther disposable word index. Which will be destroyed by this class at any needed moment, so no one else must have a reference or a pointer to the argument object

Parameters

memory_factor	the memory factor for the SubWordIndexType constructor

Definition at line 94 of file optimizing_word_index.hpp.

8.48.2.2 template<typename sub_word_index_type > virtual uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type >::~optimizing_word_index ( ) [inline], [virtual]

The basic destructor

Definition at line 255 of file optimizing_word_index.hpp.

8.48.3 Member Function Documentation

8.48.3.1 template<typename sub_word_index_type > void uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type >::count_word (const text_piece_reader & *word,* prob_weight *prob*) [inline]

This method is to be used when the word counting is needed.

See also

AWordIndex

Definition at line 196 of file optimizing_word_index.hpp.

8.48.3.2 template<typename sub_word_index_type > void uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type >::do_post_actions ( ) [inline]

Is to be called if the post actions are needed right after that all the individual words have beed added into the index.

See also

AWordIndex

Definition at line 224 of file optimizing_word_index.hpp.

8.48.3.3 template<typename sub_word_index_type > void uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type >::do_post_word_count ( ) [inline]

Should be called if the word count is needed after all the words have been counted.

See also

AWordIndex

Definition at line 205 of file optimizing_word_index.hpp.

8.48.3.4 template<typename sub_word_index_type > size_t uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type >::get_number_of_words (const size_t *num_words*) const [inline]

Allows to get the total words count including the unknown and undefined words

See also

AWordIndex

Definition at line 129 of file optimizing_word_index.hpp.

8.48.3.5 template<typename sub_word_index_type > word_uid uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type >::get_word_id (const text_piece_reader & *token*) const [inline]

This function gets an id for the given word word based no the stored 1-Grams.

See also

AWordIndex

Definition at line 146 of file optimizing_word_index.hpp.

8.48.3.6 template<typename sub_word_index_type > bool uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type >::is_post_actions_needed ( ) const [inline]

Indicates if the post-actions are needed. The post actions should be called after all the words have been filled into the index.

See also

AWordIndex

Definition at line 215 of file optimizing_word_index.hpp.

8.48.3.7 template<typename sub_word_index_type > bool uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type >::is_word_counts_needed ( ) const [inline]

This method allows to indicate whether word counting is needed by the given implementation of the word index.

See also

AWordIndex

Definition at line 188 of file optimizing_word_index.hpp.

8.48.3.8 template<typename sub_word_index_type > static constexpr bool uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type >::is_word_index_continuous ( ) [inline], [static]

Allows to indicate if the word index is continuous, i.e. it issues the word ids in a continuous range starting from 0.

See also

AWordIndex

Returns

true - this word index is continuous.

Definition at line 248 of file optimizing_word_index.hpp.

8.48.3.9 template<typename sub_word_index_type > bool uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type >::is_word_registering_needed ( ) const [inline]

This method allows to indicate whether registering a word is needed by the given implementation of the word index.

See also

AWordIndex

Definition at line 171 of file optimizing_word_index.hpp.

8.48.3.10 template<typename sub_word_index_type > word_uid uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type >::register_word (const text_piece_reader & *token*) [inline]

This function creates/gets an id for the given word.

See also

AWordIndex

Definition at line 179 of file optimizing_word_index.hpp.

8.48.3.11 template<typename sub_word_index_type > void uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type >::reserve (const size_t *num_words*) [inline]

This method should be used to pre-allocate the word index

Parameters

num_words	the number of words

Definition at line 105 of file optimizing_word_index.hpp.

The documentation for this class was generated from the following file:

- inc/server/lm/dictionaries/optimizing_word_index.hpp

8.49 uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_LENGTH > Class Template Reference

`#include <m_gram_payload.hpp>`

Public Types

- typedef Byte_M_Gram_Id< word_uid > m_gram_id_type

Public Member Functions

- phrase_base (word_uid *word_ids, phrase_length actual_level)
- phrase_base ()
- phrase_length get_num_words () const
- word_uid get_last_word_id () const
- phrase_length get_first_word_idx () const
- phrase_length get_last_word_idx () const
- const word_uid * word_ids () const
- word_uid operator[ ] (const phrase_length word_idx) const
- uint8_t create_phrase_id (const phrase_length begin_word_idx, const phrase_length number_of_words, TM_Gram_Id_Value_Ptr &p_m_gram_id) const
- const TM_Gram_Id_Value_Ptr get_phrase_id_ref (const phrase_length begin_word_idx, const phrase_length number_of_words, uint8_t &len_bytes)

Protected Member Functions

- void set_word_ids (const phrase_length num_words, const word_uid *word_ids)

8.49.1 Detailed Description

template<phrase_length MAX_PHRASE_LENGTH, phrase_length MAX_PHRASE_ID_LENGTH>class uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_LENGTH >

This class is the base class for all the M-gram classes used

Definition at line 86 of file m_gram_payload.hpp.

8.49.2 Member Typedef Documentation

8.49.2.1 template<phrase_length MAX_PHRASE_LENGTH, phrase_length MAX_PHRASE_ID_LENGTH> typedef Byte_M_Gram_Id<word_uid> uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_LENGTH >::m_gram_id_type

Definition at line 89 of file m_gram_payload.hpp.

8.49.3 Constructor & Destructor Documentation

8.49.3.1 template<phrase_length MAX_PHRASE_LENGTH, phrase_length MAX_PHRASE_ID_LENGTH> uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_LENGTH >::phrase_base (word_uid * *word_ids,* phrase_length *actual_level*) [inline]

The basic constructor, is to be used when the M-gram level is known beforehand. Allows to set the actual M-gram level to a concrete value.

Parameters

word_ids	the pointer to the word ids array to store NOTE: this pointer must remain through out the lifetime of the object, unless re-set by the appropriate method
actual_level	the actual level of the m-gram that will be used should be <= M_GRAM_LENGTH

Definition at line 100 of file m_gram_payload.hpp.

8.49.3.2 template<phrase_length MAX_PHRASE_LENGTH, phrase_length MAX_PHRASE_ID_LENGTH> uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_LENGTH >::phrase_base ( ) [inline]

The basic constructor, is to be used when the phrase will actual level is not known beforehand - used e.g. in the query m-gram sub-class. The actual m-gram level is set to be undefined. Filling in the phrase tokens is done elsewhere.

Parameters

word_index	the used word index

Definition at line 120 of file m_gram_payload.hpp.

8.49.4 Member Function Documentation

8.49.4.1 template<phrase_length MAX_PHRASE_LENGTH, phrase_length MAX_PHRASE_ID_LENGTH> uint8_t uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_LENGTH >::create_phrase_id (const phrase_length *begin_word_idx,* const phrase_length *number_of_words,* TM_Gram_Id_Value_Ptr & *p_m_gram_id*) const [inline]

Allows to create a new m-gram id for the sub-hrase defined by the given of the method template parameters. For the argument reference to the id data pointer the following holds: a) If there was no memory allocated for the M-gram

id then there will be allocated as much as needed to store the given id. b) If there was memory allocated then no re-allocation will be done, then it is assumed that enough memory was allocated

Parameters

begin_word_idx	the index of the first word in the sub-m-gram, indexes start with 0
number_of_words	the number of sub-m-gram words
word_ids	the list of the word ids for the entire m-gram, where at least the m-gram word ids for the sub-m-gram defined by the template parameters are known and initialized.
p_m_gram_id	the reference to the M-gram id data pointer to be initialized with the M-gram id data, must be pre-allocated

Definition at line 192 of file m_gram_payload.hpp.

8.49.4.2 template<phrase_length MAX_PHRASE_LENGTH, phrase_length MAX_PHRASE_ID_LENGTH> phrase_length uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_LENGTH >::get_first_word_idx ( ) const [inline]

Allows to retrieve the actual begin word index

Returns

the index of the begin word

Definition at line 146 of file m_gram_payload.hpp.

8.49.4.3 template<phrase_length MAX_PHRASE_LENGTH, phrase_length MAX_PHRASE_ID_LENGTH> word_uid uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_LENGTH >::get_last_word_id ( ) const [inline]

Allows to retrieve the actual end word id of the m-gram

Returns

the id of the last word

Definition at line 138 of file m_gram_payload.hpp.

8.49.4.4 template<phrase_length MAX_PHRASE_LENGTH, phrase_length MAX_PHRASE_ID_LENGTH> phrase_length uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_LENGTH >::get_last_word_idx ( ) const [inline]

Allows to retrieve the actual end word index

Returns

the index of the end word

Definition at line 154 of file m_gram_payload.hpp.

8.49.4.5 template<phrase_length MAX_PHRASE_LENGTH, phrase_length MAX_PHRASE_ID_LENGTH> phrase_length uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_LENGTH >::get_num_words ( ) const [inline]

Allows to obtain the actual m-gram level

Returns

the actual m-gram level

Definition at line 130 of file m_gram_payload.hpp.

8.49.4.6 template<phrase_length MAX_PHRASE_LENGTH, phrase_length MAX_PHRASE_ID_LENGTH> const TM_Gram_Id_Value_Ptr uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_LENGTH >::get_phrase_id_ref (const phrase_length *begin_word_idx,* const phrase_length *number_of_words,* uint8_t & *len_bytes*) [inline]

Allows to create a new m-gram id for the sub-phrase defined by the given of the method template parameters. For the argument reference to the id data pointer the following holds: a) If there was no memory allocated for the M-gram id then there will be allocated as much as needed to store the given id. b) If there was memory allocated then no re-allocation will be done, then it is assumed that enough memory was allocated

Parameters

begin_word_idx	the index of the first word in the sub-m-gram, indexes start with 0
number_of_words	the number of sub-m-gram words
word_ids	the list of the word ids for the entire m-gram, where at least the m-gram word ids for the sub-m-gram defined by the template parameters are known and initialized.
p_m_gram_id	the reference to the M-gram id data pointer to be initialized with the M-gram id data, must be pre-allocated

Definition at line 225 of file m_gram_payload.hpp.

8.49.4.7 template<phrase_length MAX_PHRASE_LENGTH, phrase_length MAX_PHRASE_ID_LENGTH> word_uid uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_LENGTH >::operator[ ] (const phrase_length *word_idx*) const [inline]

Allows get the word id for the given word index

Parameters

word_idx	the word index

Returns

the word id

Definition at line 172 of file m_gram_payload.hpp.

8.49.4.8 template<phrase_length MAX_PHRASE_LENGTH, phrase_length MAX_PHRASE_ID_LENGTH> void uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_LENGTH >::set_word_ids (const phrase_length *num_words,* const word_uid * *word_ids*) [inline], [protected]

Allows to set the pointer to the word ids

Parameters

word_ids	the pointer to the void ids
num_words	the number of words in the array

Definition at line 252 of file m_gram_payload.hpp.

8.49.4.9 template<phrase_length MAX_PHRASE_LENGTH, phrase_length MAX_PHRASE_ID_LENGTH> const word_uid* uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_LENGTH >::word_ids ( ) const [inline]

Allows to work with the list of ids as with the continuous array. This function retrieves the pointer to the last word id of the m-gram.

Returns

the pointer to the first word id element,

Definition at line 163 of file m_gram_payload.hpp.

The documentation for this class was generated from the following file:

- inc/server/lm/mgrams/m_gram_payload.hpp

8.50 uva::smt::bpbd::server::decoder::sentence::phrase_data_entry Struct Reference

`#include <sentence_data_map.hpp>`

Public Member Functions

- phrase_data_entry ()
- ~phrase_data_entry ()

Public Attributes

- uint32_t m_begin_ch_idx
- uint32_t m_end_ch_idx
- phrase_uid m_phrase_uid
- tm_const_source_entry_ptr m_source_entry
- prob_weight future_cost

8.50.1 Detailed Description

This structure stores the source phrase information data. This data is the begin and end character position of the phrase in the original sentence, also the first and the last word indexes, the phrase id and the available translation, i.e. source entry.

Definition at line 67 of file sentence_data_map.hpp.

8.50.2 Constructor & Destructor Documentation

8.50.2.1 uva::smt::bpbd::server::decoder::sentence::phrase_data_entry::phrase_data_entry ( ) [inline]

The basic constructor, does default initialization of the structure fields

Definition at line 72 of file sentence_data_map.hpp.

8.50.2.2 uva::smt::bpbd::server::decoder::sentence::phrase_data_entry::~phrase_data_entry ( ) [inline]

The basic destructor

Definition at line 80 of file sentence_data_map.hpp.

8.50.3 Member Data Documentation

8.50.3.1 prob_weight uva::smt::bpbd::server::decoder::sentence::phrase_data_entry::future_cost

Definition at line 95 of file sentence_data_map.hpp.

8.50.3.2 uint32_t uva::smt::bpbd::server::decoder::sentence::phrase_data_entry::m_begin_ch_idx

Definition at line 84 of file sentence_data_map.hpp.

8.50.3.3 uint32_t uva::smt::bpbd::server::decoder::sentence::phrase_data_entry::m_end_ch_idx

Definition at line 86 of file sentence_data_map.hpp.

8.50.3.4 phrase_uid uva::smt::bpbd::server::decoder::sentence::phrase_data_entry::m_phrase_uid

Definition at line 89 of file sentence_data_map.hpp.

8.50.3.5 tm_const_source_entry_ptr uva::smt::bpbd::server::decoder::sentence::phrase_data_entry::m_source_entry

Definition at line 92 of file sentence_data_map.hpp.

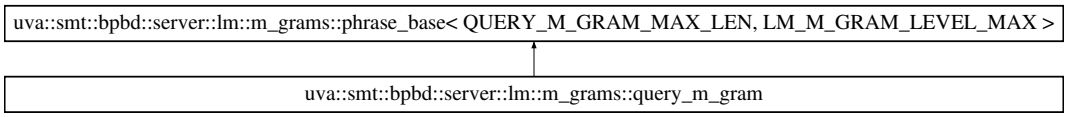
The documentation for this struct was generated from the following file:

- inc/server/decoder/sentence/sentence_data_map.hpp

8.51 uva::smt::bpbd::server::lm::m_grams::query_m_gram Class Reference

`#include <query_m_gram.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::m_grams::query_m_gram:



Public Types

- typedef phrase_base< QUERY_M_GRAM_MAX_LEN, LM_M_GRAM_LEVEL_MAX > BASE

Public Member Functions

- query_m_gram ()
- uint64_t get_hash (phrase_length begin_word_idx, const phrase_length end_word_idx) const
- void set_m_gram (const phrase_length num_words, const word_uid *word_ids)

Friends

- ostream & operator<< (ostream &stream, const query_m_gram &gram)

Additional Inherited Members

8.51.1 Detailed Description

This class is used to represent the N-Gram that will be queried against the language model.

Definition at line 56 of file query_m_gram.hpp.

8.51.2 Member Typedef Documentation

8.51.2.1 typedef phrase_base<QUERY_M_GRAM_MAX_LEN, LM_M_GRAM_LEVEL_MAX> uva::smt::bpbd::server::lm::m_grams::query_m_gram::BASE

Definition at line 59 of file query_m_gram.hpp.

8.51.3 Constructor & Destructor Documentation

8.51.3.1 uva::smt::bpbd::server::lm::m_grams::query_m_gram::query_m_gram ( ) [inline]

The basic constructor, is to be used when the M-gram will actual level is not known beforehand - used e.g. in the query m-gram sub-class. The actual m-gram level is set to be undefined. Filling in the M-gram tokens is done elsewhere.

Parameters

word_index	the used word index

Definition at line 68 of file query_m_gram.hpp.

8.51.4 Member Function Documentation

8.51.4.1 uint64_t uva::smt::bpbd::server::lm::m_grams::query_m_gram::get_hash (phrase_length *begin_word_idx,* const phrase_length *end_word_idx*) const [inline]

Allows to retrieve the hash value for the sub-m-gram defined by the parameters

Parameters

begin_word_idx	the begin word index of the sub-m-gram
end_word_idx	the end word index of the sub-m-gram

Returns

the hash value for the given sub-m-gram

Definition at line 78 of file query_m_gram.hpp.

8.51.4.2 void uva::smt::bpbd::server::lm::m_grams::query_m_gram::set_m_gram (const phrase_length *num_words,* const word_uid * *word_ids*) [inline]

Tokenise a given piece of text into a space separated list of text pieces.

Parameters

text	the piece of text to tokenise
gram	the gram container to put data into

Definition at line 137 of file query_m_gram.hpp.

8.51.5 Friends And Related Function Documentation

8.51.5.1 ostream& operator<< (ostream & *stream,* const query_m_gram & *gram*) [friend]

Allows to serialize the m-gram to the output stream as a string

Parameters

stream	the reference to the stream to output into
gram	the m-gram object to output

Returns

the reference to the stream

Definition at line 44 of file query_m_gram.cpp.

The documentation for this class was generated from the following file:

- inc/server/lm/mgrams/query_m_gram.hpp

8.52 uva::utils::containers::alloc::greedy_memory_allocator< T >::rebind< U > Struct Template Reference

`#include <greedy_memory_allocator.hpp>`

Public Types

- typedef greedy_memory_allocator< U > other

8.52.1 Detailed Description

template<typename T>template<typename U>struct uva::utils::containers::alloc::greedy_memory_allocator< T >::rebind< U >

Definition at line 144 of file greedy_memory_allocator.hpp.

8.52.2 Member Typedef Documentation

8.52.2.1 template<typename T> template<typename U > typedef greedy_memory_allocator<U> uva::utils::containers::alloc::greedy_memory_allocator< T >::rebind< U >::other

Definition at line 145 of file greedy_memory_allocator.hpp.

The documentation for this struct was generated from the following file:

- inc/common/utils/containers/greedy_memory_allocator.hpp

8.53 uva::smt::bpbd::server::rm::builders::rm_basic_builder< model_type, reader_type > Class Template Reference

`#include <rm_basic_builder.hpp>`

Public Member Functions

- rm_basic_builder (const rm_parameters &params, model_type &model, reader_type &reader)
- void build ()

Protected Member Functions

- void process_entry_weights (text_piece_reader &rest, rm_entry &entry)
- template<bool count_or_build>
  void parse_rm_file (tm_query_proxy &query)
- void count_source_target_phrases (tm_query_proxy &query)
- void process_source_entries (tm_query_proxy &query)

8.53.1 Detailed Description

template<typename model_type, typename reader_type>class uva::smt::bpbd::server::rm::builders::rm_basic_builder< model_type, reader_type >

This class represents a basic reader of the reordering model. It allows to read a text-formatted reordering model and to put it into the given instance of the model class. It assumes the simple text model format as used by Oyster or Moses. See http://www.statmt.org/moses/?n=Moses.Tutorial for some info. The reordering model is also commonly known as a phrase table.

Definition at line 75 of file rm_basic_builder.hpp.

8.53.2 Constructor & Destructor Documentation

8.53.2.1 template<typename model_type , typename reader_type > uva::smt::bpbd::server::rm::builders::rm_basic_builder< model_type, reader_type >::rm_basic_builder (const rm_parameters & *params,* model_type & *model,* reader_type & *reader*) [inline]

The basic constructor of the builder object params the model parameters

Parameters

model	the model to put the data into
reader	the reader to read the data from

Definition at line 84 of file rm_basic_builder.hpp.

8.53.3 Member Function Documentation

8.53.3.1 template<typename model_type , typename reader_type > void uva::smt::bpbd::server::rm::builders::rm_basic_builder< model_type, reader_type >::build ( ) [inline]

Allows to build the model by reading from the reader object. This is a two step process as first we need the number of distinct source phrases.

Definition at line 93 of file rm_basic_builder.hpp.

8.53.3.2 template<typename model_type , typename reader_type > void uva::smt::bpbd::server::rm::builders::rm_basic_builder< model_type, reader_type >::count_source_target_phrases (tm_query_proxy & *query*) [inline], [protected]

Allows to count and set the number of source phrases

Parameters

count_or_build	if true then we do count entries if false then we do build be model
query	the translation model query object to query the translation model for present entries

Definition at line 232 of file rm_basic_builder.hpp.

8.53.3.3 template<typename model_type , typename reader_type > template<bool count_or_build> void uva::smt::bpbd::server::rm::builders::rm_basic_builder< model_type, reader_type >::parse_rm_file (tm_query_proxy & *query*) [inline], [protected]

Allows to parse the RM model file and do two things depending on the value of the template parameter:

1. Count the number of valid entries
2. Build the RM model NOTE: This two pass parsing is not optimal but we have to do it as we need to know the number of valid entries beforehand, an optimization might be needed!

Parameters

count_or_build	if true then count if false then build
query	the TM query to check if the source/taret are known

Definition at line 146 of file rm_basic_builder.hpp.

8.53.3.4 template<typename model_type , typename reader_type > void uva::smt::bpbd::server::rm::builders::rm_basic_builder< model_type, reader_type >::process_entry_weights (text_piece_reader & *rest,* rm_entry & *entry*) [inline], [protected]

Allows to parse the reordering weights and set them into the reordering entry

Parameters

rest	the line to be parsed, starts with a space
entry	the entry to put the values into

Definition at line 116 of file rm_basic_builder.hpp.

8.53.3.5 template<typename model_type , typename reader_type > void uva::smt::bpbd::server::rm::builders::rm_basic_builder< model_type, reader_type >::process_source_entries (tm_query_proxy & *query*) [inline], [protected]

Allows to process translations.

Parameters

query	the translation model query object to query the translation model for present entries

Definition at line 255 of file rm_basic_builder.hpp.

The documentation for this class was generated from the following file:

- inc/server/rm/builders/rm_basic_builder.hpp

8.54 uva::smt::bpbd::server::rm::models::rm_basic_model Class Reference

`#include <rm_basic_model.hpp>`

Public Types

- typedef fixed_size_hashmap< rm_entry, const phrase_uid & > rm_entry_map

Public Member Functions

- rm_basic_model ()
- ~rm_basic_model ()
- bool is_num_entries_needed () const
- void set_num_entries (size_t num_entries)
- rm_entry & add_entry (const phrase_uid &source_uid, const phrase_uid &target_uid)
- void find_unk_entry ()
- void find_begin_end_entries ()
- const rm_entry * get_begin_tag_entry () const
- const rm_entry * get_end_tag_entry () const
- bool is_unk_entry (const rm_entry *entry) const
- const rm_entry * get_entry (const phrase_uid uid) const
- const rm_entry * get_entry (const phrase_uid &source_uid, const phrase_uid &target_uid) const
- void log_model_type_info ()

Public Attributes

- const phrase_uid SOURCE_UNK_UID
- const phrase_uid TARGET_UNK_UID
- const phrase_uid BEGIN_SENT_TAG_UID
- const phrase_uid END_SENT_TAG_UID

8.54.1 Detailed Description

This class represents a basic reordering model implementation. The basic model is based on the fixed size hash map which is a self-implemented linear probing hash map also used in several tries. This basic model also does not store the phrases as is but rather the hash values thereof. So it is a hash based implementation which reduces memory but might occasionally result in collisions, the latter is not checked.

Definition at line 62 of file rm_basic_model.hpp.

8.54.2 Member Typedef Documentation

8.54.2.1 typedef fixed_size_hashmap<rm_entry, const phrase_uid &> uva::smt::bpbd::server::rm::models::rm_basic_model::rm_entry_map

Definition at line 74 of file rm_basic_model.hpp.

8.54.3 Constructor & Destructor Documentation

8.54.3.1 uva::smt::bpbd::server::rm::models::rm_basic_model::rm_basic_model ( ) [inline]

The basic class constructor

Definition at line 79 of file rm_basic_model.hpp.

8.54.3.2 uva::smt::bpbd::server::rm::models::rm_basic_model::~rm_basic_model ( ) [inline]

The basic destructor

Definition at line 94 of file rm_basic_model.hpp.

8.54.4 Member Function Documentation

8.54.4.1 rm_entry& uva::smt::bpbd::server::rm::models::rm_basic_model::add_entry (const phrase_uid & *source_uid,* const phrase_uid & *target_uid*) [inline]

Allows to add a new reordering entry to the model

Parameters

source_uid	the source phrase uid
target_uid	the target phrase uid

Returns

the reference to the newly allocated entry

Definition at line 129 of file rm_basic_model.hpp.

8.54.4.2 void uva::smt::bpbd::server::rm::models::rm_basic_model::find_begin_end_entries ( ) [inline]

Allows to get the reordering model entry for the given tag

Parameters

tag	the tag to get the reordering entry for
tag_entry	[out] the reordering model entry pointer reference to be set

Definition at line 163 of file rm_basic_model.hpp.

8.54.4.3 void uva::smt::bpbd::server::rm::models::rm_basic_model::find_unk_entry ( ) [inline]

This method must be called after the model is loaded in order to find the UNK/UNK phrase entry

Definition at line 146 of file rm_basic_model.hpp.

8.54.4.4 const rm_entry* uva::smt::bpbd::server::rm::models::rm_basic_model::get_begin_tag_entry ( ) const [inline]

Allows to get the sentence begin tag entry if found

Returns

the sentence begin tag entry if found or NULL

Definition at line 185 of file rm_basic_model.hpp.

8.54.4.5 const rm_entry* uva::smt::bpbd::server::rm::models::rm_basic_model::get_end_tag_entry ( ) const [inline]

Allows to get the sentence end tag entry if found

Returns

the sentence end tag entry if found or NULL

Definition at line 193 of file rm_basic_model.hpp.

8.54.4.6 const rm_entry* uva::smt::bpbd::server::rm::models::rm_basic_model::get_entry (const phrase_uid *uid*) const [inline]

Allows to get the reordering entry for the given source/target pair the latter is identified with a phrase id. In case the entry is not present we return the data for the UNK/UNK pair.

Parameters

uid	the source/target phrase pair uid

Returns

the reordering entry, always NOT NULL!

Definition at line 213 of file rm_basic_model.hpp.

8.54.4.7 const rm_entry* uva::smt::bpbd::server::rm::models::rm_basic_model::get_entry (const phrase_uid & *source_uid,* const phrase_uid & *target_uid*) const [inline]

Allows to get the reordering entry for the given source/target pair In case the entry is not present we return the data for the UNK/UNK pair.

Parameters

source_uid	the source phrase uid
target_uid	the target phrase uid

Returns

the reordering entry, always NOT NULL!

Definition at line 237 of file rm_basic_model.hpp.

8.54.4.8 bool uva::smt::bpbd::server::rm::models::rm_basic_model::is_num_entries_needed ( ) const [inline]

This method allows to detect if the number of reordering entries is needed before the entries are being added.

Returns

true as this model type uses filed-size hash maps

Definition at line 108 of file rm_basic_model.hpp.

8.54.4.9 bool uva::smt::bpbd::server::rm::models::rm_basic_model::is_unk_entry (const rm_entry * *entry*) const [inline]

Allows to detect whether the given entry is an entry for the unknown phrase pair

Parameters

entry	the entry pointer to be checked for being from the UNK/UNK entry

Returns

true if this entry is for the unknown phrase pair, otherwise false

Definition at line 202 of file rm_basic_model.hpp.

8.54.4.10 void uva::smt::bpbd::server::rm::models::rm_basic_model::log_model_type_info ( ) [inline]

Allows to log the model type info

Definition at line 248 of file rm_basic_model.hpp.

8.54.4.11 void uva::smt::bpbd::server::rm::models::rm_basic_model::set_num_entries (size_t *num_entries*) [inline]

This method is needed to set the number of reordering entries in the model.

Parameters

num_entries	the number of entries in the reordering model

Definition at line 116 of file rm_basic_model.hpp.

8.54.5 Member Data Documentation

8.54.5.1 const phrase_uid uva::smt::bpbd::server::rm::models::rm_basic_model::BEGIN_SENT_TAG_UID

Definition at line 69 of file rm_basic_model.hpp.

8.54.5.2 const phrase_uid uva::smt::bpbd::server::rm::models::rm_basic_model::END_SENT_TAG_UID

Definition at line 71 of file rm_basic_model.hpp.

8.54.5.3 const phrase_uid uva::smt::bpbd::server::rm::models::rm_basic_model::SOURCE_UNK_UID

Definition at line 65 of file rm_basic_model.hpp.

8.54.5.4 const phrase_uid uva::smt::bpbd::server::rm::models::rm_basic_model::TARGET_UNK_UID

Definition at line 67 of file rm_basic_model.hpp.

The documentation for this class was generated from the following file:

- inc/server/rm/models/rm_basic_model.hpp

8.55 uva::smt::bpbd::server::rm::rm_configurator Class Reference

`#include <rm_configurator.hpp>`

Static Public Member Functions

- static void connect (const rm_parameters &params)
- static void disconnect ()
- static rm_query_proxy & allocate_query_proxy ()
- static void dispose_query_proxy (rm_query_proxy &query)

8.55.1 Detailed Description

This class represents a singleton that allows to configure the reordering model and then issue a proxy object for performing the queries against it.

Definition at line 52 of file rm_configurator.hpp.

8.55.2 Member Function Documentation

8.55.2.1 static rm_query_proxy& uva::smt::bpbd::server::rm::rm_configurator::allocate_query_proxy ( ) [inline], [static]

Allows to return an instance of the query proxy, is to be returned by calling the dispose method.

Returns

an instance of the query executor.

Definition at line 91 of file rm_configurator.hpp.

8.55.2.2 static void uva::smt::bpbd::server::rm::rm_configurator::connect (const rm_parameters & *params*) [inline], [static]

This method allows to connect to the reordering model. This method is to be called only once! The latter is not checked but is a must.

Parameters

params	the reordering model parameters to be set, this class only stores the referent to the parameters.

Definition at line 62 of file rm_configurator.hpp.

8.55.2.3 static void uva::smt::bpbd::server::rm::rm_configurator::disconnect ( ) [inline], [static]

Allows to disconnect from the reordering model.

Definition at line 76 of file rm_configurator.hpp.

8.55.2.4 static void uva::smt::bpbd::server::rm::rm_configurator::dispose_query_proxy (rm_query_proxy & *query*) [inline], [static]

Dispose the previously allocated query object

Parameters

query	the query to dispose

Definition at line 102 of file rm_configurator.hpp.

The documentation for this class was generated from the following files:

- inc/server/rm/rm_configurator.hpp
- src/server/rm/rm_configurator.cpp

8.56 uva::smt::bpbd::server::rm::models::rm_entry_temp< num_features > Class Template Reference

`#include <rm_entry.hpp>`

Public Member Functions

- rm_entry_temp ()
- ~rm_entry_temp ()
- const prob_weight * get_weights () const
- template<bool is_from>
  const prob_weight get_weight (const reordering_orientation orient) const
- prob_weight & operator[ ] (size_t idx)
- void set_entry_uid (const phrase_uid &uid)
- bool operator== (const phrase_uid &uid) const
- bool operator== (const rm_entry_temp &other) const

Static Public Attributes

- static constexpr uint8_t NUM_FEATURES = num_features

Friends

- template<uint8_t num_weights>
  ostream & operator<< (ostream &stream, const rm_entry_temp< num_weights > &entry)

8.56.1 Detailed Description

template<uint8_t num_features>class uva::smt::bpbd::server::rm::models::rm_entry_temp< num_features >

This is the reordering entry class it stores the reordering penalties for one source to target phrase.

Parameters

num_features	is the number of reordering weights

Definition at line 68 of file rm_entry.hpp.

8.56.2 Constructor & Destructor Documentation

8.56.2.1 template<uint8_t num_features> uva::smt::bpbd::server::rm::models::rm_entry_temp< num_features >::rm_entry_temp ( ) [inline]

The basic constructor

Definition at line 76 of file rm_entry.hpp.

8.56.2.2 template<uint8_t num_features> uva::smt::bpbd::server::rm::models::rm_entry_temp< num_features >::~rm_entry_temp ( ) [inline]

The basic destructor

Definition at line 83 of file rm_entry.hpp.

8.56.3 Member Function Documentation

8.56.3.1 template<uint8_t num_features> template<bool is_from> const prob_weight uva::smt::bpbd::server::rm::models::rm_entry_temp< num_features >::get_weight (const reordering_orientation *orient*) const [inline]

Allows to get the weight for the given distortion value

Parameters

is_from	the flag allowing to distinguish between the from and to case if true then we get the value from the from source phrase case if false then we get the value for the to source phrase case
orient	the reordering orientation

Returns

the weight for the given distortion value

Definition at line 103 of file rm_entry.hpp.

8.56.3.2 template<uint8_t num_features> const prob_weight* uva::smt::bpbd::server::rm::models::rm_entry_temp< num_features >::get_weights ( ) const [inline]

Allows to get the entry weights array

Returns

the entry weights array

Definition at line 90 of file rm_entry.hpp.

8.56.3.3 template<uint8_t num_features> bool uva::smt::bpbd::server::rm::models::rm_entry_temp< num_features >::operator== (const phrase_uid & *uid*) const [inline]

The comparison operator, allows to compare entries

Parameters

phrase_uid	the unique identifier of the source/target phrase pair entry to compare with

Returns

true if the provided uid is equal to the uid of this entry, otherwise false

Definition at line 161 of file rm_entry.hpp.

8.56.3.4 template<uint8_t num_features> bool uva::smt::bpbd::server::rm::models::rm_entry_temp< num_features >::operator== (const rm_entry_temp< num_features > & *other*) const [inline]

The comparison operator, allows to compare entries

Parameters

other	the other entry to compare with

Returns

true if the provided entry has the same uid as this one, otherwise false

Definition at line 170 of file rm_entry.hpp.

8.56.3.5 template<uint8_t num_features> prob_weight& uva::smt::bpbd::server::rm::models::rm_entry_temp< num_features >::operator[ ] (size_t *idx*) [inline]

This operator allows to work with the given reordering entry weights in an array fashion

Parameters

idx	the index of the feature

Returns

the feature value

Definition at line 139 of file rm_entry.hpp.

8.56.3.6 template<uint8_t num_features> void uva::smt::bpbd::server::rm::models::rm_entry_temp< num_features >::set_entry_uid (const phrase_uid & *uid*) [inline]

Allows to set the unique source target entry identifier

Parameters

uid	the unique identifier of the source/target entry

Definition at line 152 of file rm_entry.hpp.

8.56.4 Friends And Related Function Documentation

8.56.4.1 template<uint8_t num_features> template<uint8_t num_weights> ostream& operator<< (ostream & *stream*, const rm_entry_temp< num_weights > & *entry*) [friend]

This operator allows to stream the reordering entry to the output stream

Parameters

stream	the stream to send the data into
entry	the entry to stream

Returns

the reference to the same stream is returned

Definition at line 198 of file rm_entry.hpp.

8.56.5 Member Data Documentation

8.56.5.1 template<uint8_t num_features> constexpr uint8_t uva::smt::bpbd::server::rm::models::rm_entry_temp< num_features >::NUM_FEATURES = num_features [static]

Definition at line 71 of file rm_entry.hpp.

The documentation for this class was generated from the following file:

- inc/server/rm/models/rm_entry.hpp

8.57 uva::smt::bpbd::server::rm::rm_parameters Struct Reference

`#include <rm_parameters.hpp>`

Public Member Functions

- void finalize ()

Public Attributes

- string m_conn_string
- size_t m_num_lambdas
- float m_lambdas [NUM_RM_FEATURES]

8.57.1 Detailed Description

This structure stores the reordering model parameters

Definition at line 53 of file rm_parameters.hpp.

8.57.2 Member Function Documentation

8.57.2.1 void uva::smt::bpbd::server::rm::rm_parameters::finalize ( ) [inline]

Allows to verify the parameters to be correct.

Definition at line 66 of file rm_parameters.hpp.

8.57.3 Member Data Documentation

8.57.3.1 string uva::smt::bpbd::server::rm::rm_parameters::m_conn_string

Definition at line 55 of file rm_parameters.hpp.

8.57.3.2 float uva::smt::bpbd::server::rm::rm_parameters::m_lambdas[NUM_RM_FEATURES]

Definition at line 61 of file rm_parameters.hpp.

8.57.3.3 size_t uva::smt::bpbd::server::rm::rm_parameters::m_num_lambdas

Definition at line 58 of file rm_parameters.hpp.

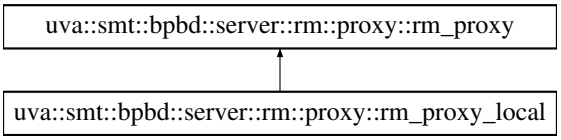
The documentation for this struct was generated from the following file:

- inc/server/rm/rm_parameters.hpp

8.58 uva::smt::bpbd::server::rm::proxy::rm_proxy Class Reference

`#include <rm_proxy.hpp>`

Inheritance diagram for uva::smt::bpbd::server::rm::proxy::rm_proxy:



Public Member Functions

- virtual void connect (const rm_parameters &params)=0
- virtual void disconnect ()=0
- virtual ~rm_proxy ()
- virtual rm_query_proxy & allocate_query_proxy ()=0
- virtual void dispose_query_proxy (rm_query_proxy &query)=0

8.58.1 Detailed Description

This is the reordering model proxy interface class it allows to interact with any sort of local and remote models in a uniform way.

Definition at line 42 of file rm_proxy.hpp.

8.58.2 Constructor & Destructor Documentation

8.58.2.1 virtual uva::smt::bpbd::server::rm::proxy::rm_proxy::~rm_proxy ( ) [inline], [virtual]

The basic virtual destructor

Definition at line 59 of file rm_proxy.hpp.

8.58.3 Member Function Documentation

8.58.3.1 virtual rm_query_proxy& uva::smt::bpbd::server::rm::proxy::rm_proxy::allocate_query_proxy ( ) [pure virtual]

This method allows to get a query executor for the given trie

Returns

the trie query proxy object

Implemented in uva::smt::bpbd::server::rm::proxy::rm_proxy_local.

8.58.3.2 virtual void uva::smt::bpbd::server::rm::proxy::rm_proxy::connect (const rm_parameters & *params*) [pure virtual]

Allows to connect to the model object based on the given parameters

Parameters

params	the model parameters

Implemented in uva::smt::bpbd::server::rm::proxy::rm_proxy_local.

8.58.3.3 virtual void uva::smt::bpbd::server::rm::proxy::rm_proxy::disconnect ( ) [pure virtual]

Allows to disconnect from the trie

Implemented in uva::smt::bpbd::server::rm::proxy::rm_proxy_local.

8.58.3.4 virtual void uva::smt::bpbd::server::rm::proxy::rm_proxy::dispose_query_proxy (rm_query_proxy & *query*) [pure virtual]

Dispose the previously allocated query object

Parameters

query	the query to dispose

Implemented in uva::smt::bpbd::server::rm::proxy::rm_proxy_local.

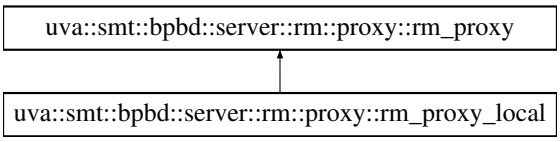
The documentation for this class was generated from the following file:

- inc/server/rm/proxy/rm_proxy.hpp

8.59 uva::smt::bpbd::server::rm::proxy::rm_proxy_local Class Reference

`#include <rm_proxy_local.hpp>`

Inheritance diagram for uva::smt::bpbd::server::rm::proxy::rm_proxy_local:



Public Member Functions

- rm_proxy_local ()
- virtual ∼rm_proxy_local ()
- virtual void connect (const rm_parameters &params)
- virtual void disconnect ()
- virtual rm_query_proxy & allocate_query_proxy ()
- virtual void dispose_query_proxy (rm_query_proxy &query)

Protected Member Functions

- template<typename rm_builder_type , typename file_reader_type >
  void load_model_data (char const *model_name, const rm_parameters &params)

8.59.1 Detailed Description

This is the reordering model proxy interface class it allows to interact with any sort of local and remote models in a uniform way.

Definition at line 59 of file rm_proxy_local.hpp.

8.59.2 Constructor & Destructor Documentation

8.59.2.1 uva::smt::bpbd::server::rm::proxy::rm_proxy_local::rm_proxy_local ( ) [inline]

The basic proxy constructor, currently does nothing except for default initialization

Definition at line 65 of file rm_proxy_local.hpp.

8.59.2.2 virtual uva::smt::bpbd::server::rm::proxy::rm_proxy_local::∼rm_proxy_local ( ) [inline], [virtual]

The basic destructor

Definition at line 71 of file rm_proxy_local.hpp.

8.59.3 Member Function Documentation

8.59.3.1 virtual rm_query_proxy& uva::smt::bpbd::server::rm::proxy::rm_proxy_local::allocate_query_proxy ( ) [inline], [virtual]

Todo {In the future we should just use a number of stack allocated objects in order to reduce the new/delete overhead}

See also

rm_proxy

Implements uva::smt::bpbd::server::rm::proxy::rm_proxy.

Definition at line 101 of file rm_proxy_local.hpp.

8.59.3.2 virtual void uva::smt::bpbd::server::rm::proxy::rm_proxy_local::connect (const rm_parameters & *params*) [inline], [virtual]

See also

rm_proxy

Implements uva::smt::bpbd::server::rm::proxy::rm_proxy.

Definition at line 79 of file rm_proxy_local.hpp.

8.59.3.3 virtual void uva::smt::bpbd::server::rm::proxy::rm_proxy_local::disconnect ( ) [inline], [virtual]

See also

rm_proxy

Implements uva::smt::bpbd::server::rm::proxy::rm_proxy.

Definition at line 92 of file rm_proxy_local.hpp.

8.59.3.4 virtual void uva::smt::bpbd::server::rm::proxy::rm_proxy_local::dispose_query_proxy (rm_query_proxy & *query*) [inline], [virtual]

Dispose the previously allocated query object

Todo {In the future we should just use a number of stack allocated objects in order to reduce the new/delete overhead}

Parameters

query	the query to dispose

Implements uva::smt::bpbd::server::rm::proxy::rm_proxy.

Definition at line 111 of file rm_proxy_local.hpp.

8.59.3.5 template<typename rm_builder_type , typename file_reader_type > void uva::smt::bpbd::server::rm::proxy::rm_proxy_local::load_model_data (char const * *model_name,* const rm_parameters & *params*) [inline], [protected]

Allows to load the model into the instance of the selected container class

Todo Add the possibility to choose between the file readers from the command line!

Parameters

the	name of the model being loaded params the model parameters

Definition at line 124 of file rm_proxy_local.hpp.

The documentation for this class was generated from the following file:

- inc/server/rm/proxy/rm_proxy_local.hpp

8.60 uva::smt::bpbd::server::rm::models::rm_query< model_type > Class Template Reference

`#include <rm_query.hpp>`

Public Types

- typedef unordered_map< phrase_uid, const rm_entry * > query_map

Public Member Functions

- rm_query (const model_type &model)
- virtual void execute (const vector< phrase_uid > &st_ids)
- ~rm_query ()
- virtual const rm_entry & get_reordering (const phrase_uid uid) const

8.60.1 Detailed Description

template<typename model_type>class uva::smt::bpbd::server::rm::models::rm_query< model_type >

This class represents a query for the reordering model

Definition at line 54 of file rm_query.hpp.

8.60.2 Member Typedef Documentation

8.60.2.1 template<typename model_type> typedef unordered_map<phrase_uid, const rm_entry *> uva::smt::bpbd::server::rm::models::rm_query< model_type >::query_map

Definition at line 59 of file rm_query.hpp.

8.60.3 Constructor & Destructor Documentation

8.60.3.1 template<typename model_type> uva::smt::bpbd::server::rm::models::rm_query< model_type >::rm_query (const model_type & *model*) [inline]

The basic constructor

Definition at line 64 of file rm_query.hpp.

8.60.3.2 template<typename model_type> uva::smt::bpbd::server::rm::models::rm_query< model_type >::~rm_query ( ) [inline]

The basic destructor

Definition at line 93 of file rm_query.hpp.

8.60.4 Member Function Documentation

8.60.4.1 template<typename model_type> virtual void uva::smt::bpbd::server::rm::models::rm_query< model_type >::execute (const vector< phrase_uid > & *st_ids*) [inline], [virtual]

Allows to execute the query, for the given source/target phrase ids

Parameters

st_ids	is the list of the source/target phrase ids for which the reordering data is needed

Definition at line 72 of file rm_query.hpp.

8.60.4.2 template<typename model_type> virtual const rm_entry& uva::smt::bpbd::server::rm::models::rm_query< model_type >::get_reordering (const phrase_uid *uid*) const [inline], [virtual]

Allows to get the source/target reordering data from the reordering model

Parameters

uid	the source/target phrase uid

Returns

the reference to the source entry, might be the one of UNK if the reordering was not found.

Definition at line 103 of file rm_query.hpp.

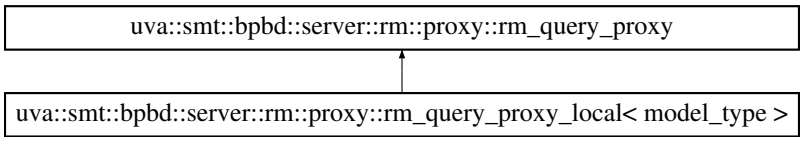
The documentation for this class was generated from the following file:

- inc/server/rm/models/rm_query.hpp

8.61 uva::smt::bpbd::server::rm::proxy::rm_query_proxy Class Reference

`#include <rm_query_proxy.hpp>`

Inheritance diagram for uva::smt::bpbd::server::rm::proxy::rm_query_proxy:



Public Member Functions

- virtual void execute (const vector< phrase_uid > &st_ids)=0
- virtual const rm_entry & get_begin_tag_reordering () const =0
- virtual const rm_entry & get_end_tag_reordering () const =0
- virtual const rm_entry & get_reordering (const phrase_uid uid) const =0
- virtual ~rm_query_proxy ()

8.61.1 Detailed Description

This class represents a reordering query proxy interface class. It allows to interact with reordering model queries in a uniform way.

Definition at line 52 of file rm_query_proxy.hpp.

8.61.2 Constructor & Destructor Documentation

8.61.2.1 virtual uva::smt::bpbd::server::rm::proxy::rm_query_proxy::~rm_query_proxy ( ) [inline],[virtual]

The basic virtual destructor

Definition at line 85 of file rm_query_proxy.hpp.

8.61.3 Member Function Documentation

8.61.3.1 virtual void uva::smt::bpbd::server::rm::proxy::rm_query_proxy::execute (const vector< phrase_uid > & *st_ids*) [pure virtual]

Allows to execute the query, for the given source/target phrase ids

Parameters

st_ids	is the list of the source/target phrase ids for which the reordering data is needed

Implemented in uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type >.

8.61.3.2 virtual const rm_entry& uva::smt::bpbd::server::rm::proxy::rm_query_proxy::get_begin_tag_reordering ( ) const [pure virtual]

Allows to retrieve the begin tag reordering entry from the reordering model

Returns

the start tag reordering entry

Implemented in uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type >.

8.61.3.3 virtual const rm_entry& uva::smt::bpbd::server::rm::proxy::rm_query_proxy::get_end_tag_reordering ( ) const [pure virtual]

Allows to retrieve the end tag reordering entry from the reordering model

Returns

the start tag reordering entry

Implemented in uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type >.

8.61.3.4 virtual const rm_entry& uva::smt::bpbd::server::rm::proxy::rm_query_proxy::get_reordering (const phrase_uid *uid*) const [pure virtual]

Allows to get the source/target reordering data from the reordering model

Parameters

uid	the source/target phrase uid

Returns

the reference to the source entry, might be the one of UNK if the reordering was not found.

Implemented in uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type >.

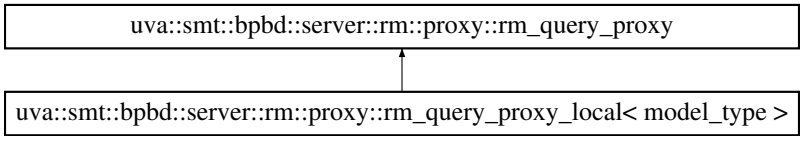
The documentation for this class was generated from the following file:

- inc/server/rm/proxy/rm_query_proxy.hpp

8.62 uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type > Class Template Reference

`#include <rm_query_proxy_local.hpp>`

Inheritance diagram for uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type >:



Public Member Functions

- rm_query_proxy_local (const model_type &model, const rm_entry &begin_tag_entry, const rm_entry &end↩ _tag_entry)
- virtual const rm_entry & get_begin_tag_reordering () const
- virtual const rm_entry & get_end_tag_reordering () const
- virtual const rm_entry & get_reordering (const phrase_uid uid) const
- virtual void execute (const vector< phrase_uid > &st_ids)
- virtual ~rm_query_proxy_local ()

8.62.1 Detailed Description

template<typename model_type>class uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type >

This is a local implementation of the reordering model query This implementation works with the local reordering model

Definition at line 48 of file rm_query_proxy_local.hpp.

8.62.2 Constructor & Destructor Documentation

8.62.2.1 template<typename model_type> uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type >::rm_query_proxy_local (const model_type & *model*, const rm_entry & *begin_tag_entry*, const rm_entry & *end_tag_entry*) [inline]

The basic constructor that accepts the reordering model reference to query to

Parameters

model	the reordering model to query
begin_tag_entry	the reference to the begin tag reordering
end_tag_entry	the reference to the end tag reordering

Definition at line 57 of file rm_query_proxy_local.hpp.

8.62.2.2 template<typename model_type> virtual uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type >::~rm_query_proxy_local ( ) [inline], [virtual]

See also

rm_query_proxy

Definition at line 97 of file rm_query_proxy_local.hpp.

8.62.3 Member Function Documentation

8.62.3.1 template<typename model_type> virtual void uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type >::execute (const vector< phrase_uid > & *st_ids*) [inline], [virtual]

See also

rm_query_proxy

Implements uva::smt::bpbd::server::rm::proxy::rm_query_proxy.

Definition at line 90 of file rm_query_proxy_local.hpp.

8.62.3.2 template<typename model_type> virtual const rm_entry& uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type >::get_begin_tag_reordering ( ) const [inline], [virtual]

Allows to retrieve the begin tag reordering entry from the reordering model

Returns

the start tag reordering entry

Implements uva::smt::bpbd::server::rm::proxy::rm_query_proxy.

Definition at line 66 of file rm_query_proxy_local.hpp.

8.62.3.3 template<typename model_type> virtual const rm_entry& uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type >::get_end_tag_reordering ( ) const [inline], [virtual]

Allows to retrieve the end tag reordering entry from the reordering model

Returns

the start tag reordering entry

Implements uva::smt::bpbd::server::rm::proxy::rm_query_proxy.

Definition at line 75 of file rm_query_proxy_local.hpp.

8.62.3.4 template<typename model_type> virtual const rm_entry& uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type >::get_reordering (const phrase_uid *uid*) const [inline], [virtual]

See also

rm_query_proxy

Implements uva::smt::bpbd::server::rm::proxy::rm_query_proxy.

Definition at line 83 of file rm_query_proxy_local.hpp.

The documentation for this class was generated from the following file:

- inc/server/rm/proxy/rm_query_proxy_local.hpp

8.63 uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_GramData< TPayloadType > Struct Template Reference

`#include <h2d_map_trie.hpp>`

Public Types

- typedef uint64_t TM_Gram_Id
- typedef S_M_GramData< TPayloadType > SELF

Public Member Functions

- S_M_GramData ()
- ~S_M_GramData ()
- bool operator== (const TM_Gram_Id &id) const

Public Attributes

- TM_Gram_Id m_id
- TPayloadType m_payload

8.63.1 Detailed Description

template<typename TPayloadType>struct uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_GramData< TPayloadType >

This template structure is used for storing trie hash map elements Each element contains and id of the m-gram and its payload - the probability/back-off data, the latter is the template parameter

Parameters

id	stores the M-gram id
payload	stores the payload which is either probability or probability with back-off

Definition at line 71 of file h2d_map_trie.hpp.

8.63.2 Member Typedef Documentation

8.63.2.1 template<typename TPayloadType > typedef S_M_GramData<TPayloadType> uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_GramData< TPayloadType >::SELF

Definition at line 75 of file h2d_map_trie.hpp.

8.63.2.2 template<typename TPayloadType > typedef uint64_t uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_GramData< TPayloadType >::TM_Gram_Id

Definition at line 73 of file h2d_map_trie.hpp.

8.63.3 Constructor & Destructor Documentation

8.63.3.1 template<typename TPayloadType > uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_GramData< TPayloadType >::S_M_GramData ( ) [inline]

The basic constructor

Definition at line 86 of file h2d_map_trie.hpp.

8.63.3.2 template<typename TPayloadType > uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_GramData< TPayloadType >::~S_M_GramData ( ) [inline]

The basic destructor

Definition at line 92 of file h2d_map_trie.hpp.

8.63.4 Member Function Documentation

8.63.4.1 template<typename TPayloadType > bool uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_GramData< TPayloadType >::operator== (const TM_Gram_Id & *id*) const [inline]

The comparison operator, allows to compare two m-gram ids

Parameters

id	the m-gram id to compare with

Returns

true if the ids are equal, otherwise false

Definition at line 101 of file h2d_map_trie.hpp.

8.63.5 Member Data Documentation

8.63.5.1 template<typename TPayloadType > TM_Gram_Id uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_GramData< TPayloadType >::m_id

Definition at line 78 of file h2d_map_trie.hpp.

8.63.5.2 template<typename TPayloadType > TPayloadType uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_GramData< TPayloadType >::m_payload

Definition at line 81 of file h2d_map_trie.hpp.

The documentation for this struct was generated from the following file:

- inc/server/lm/models/h2d_map_trie.hpp

8.64 uva::smt::bpbd::server::lm::__W2CArrayTrie::S_M_GramData< PAYLOAD_TYPE > Struct Template Reference

`#include <w2c_array_trie.hpp>`

Public Attributes

- TShortId id
- PAYLOAD_TYPE payload

Static Public Attributes

- static const mem_increase_strategy m_mem_strat

8.64.1 Detailed Description

template<typename PAYLOAD_TYPE>struct uva::smt::bpbd::server::lm::__W2CArrayTrie::S_M_GramData< PAYLOAD_TYPE >

This template structure is used for storing trie element data Each element contains a context id of the m-gram and its payload - the probability/back-off data, the latter is the template parameter

Definition at line 64 of file w2c_array_trie.hpp.

8.64.2 Member Data Documentation

8.64.2.1 template<typename PAYLOAD_TYPE > TShortId uva::smt::bpbd::server::lm::__W2CArrayTrie::S_M_GramData< PAYLOAD_TYPE >::id

Definition at line 65 of file w2c_array_trie.hpp.

8.64.2.2 template<typename PAYLOAD_TYPE > const mem_increase_strategy uva::smt::bpbd::server::lm::__W2CArrayTrie::S_M_GramData< PAYLOAD_TYPE >::m_mem_strat [static]

Initial value:

```
=
                        get_mem_incr_strat(__W2CArrayTrie::MEM_INC_TYPE,
                                __W2CArrayTrie::MIN_MEM_INC_NUM, __W2CArrayTrie::MEM_INC_FACTOR)
```

Definition at line 69 of file w2c_array_trie.hpp.

8.64.2.3 template<typename PAYLOAD_TYPE > PAYLOAD_TYPE uva::smt::bpbd::server::lm::__W2CArrayTrie::S_M_GramData< PAYLOAD_TYPE >::payload

Definition at line 66 of file w2c_array_trie.hpp.

The documentation for this struct was generated from the following file:

- inc/server/lm/models/w2c_array_trie.hpp

8.65 uva::smt::bpbd::server::lm::__G2DMapTrie::S_M_GramData< TPayloadType, TWordIdType > Struct Template Reference

```
#include <g2d_map_trie.hpp>
```

Public Types

- typedef Byte_M_Gram_Id< TWordIdType > TM_Gram_Id
- typedef S_M_GramData< TPayloadType, TWordIdType > SELF

Public Member Functions

- S_M_GramData ()
- ~S_M_GramData ()
- bool operator== (const T_Gram_Id_Key &key) const

Public Attributes

- TM_Gram_Id_Value_Ptr m_id
- TPayloadType m_payload

8.65.1 Detailed Description

template<typename TPayloadType, typename TWordIdType>struct uva::smt::bpbd::server::lm::__G2DMapTrie::S_M_Gram↩ Data< TPayloadType, TWordIdType >

This template structure is used for storing trie hash map elements Each element contains and id of the m-gram and its payload - the probability/back-off data, the latter is the template parameter

NOTE: In order to save space and increase the speed we could store key to be the hash value of the m-gram, but then we will get the h2dm trie.

Parameters

id	stores the M-gram id
payload	stores the payload which is either probability or probability with back-off

Definition at line 77 of file g2d_map_trie.hpp.

8.65.2 Member Typedef Documentation

8.65.2.1 template<typename TPayloadType , typename TWordIdType > typedef S_M_GramData<TPayloadType, TWordIdType> uva::smt::bpbd::server::lm::__G2DMapTrie::S_M_GramData< TPayloadType, TWordIdType >::SELF

Definition at line 81 of file g2d_map_trie.hpp.

8.65.2.2 template<typename TPayloadType , typename TWordIdType > typedef Byte_M_Gram_Id<TWordIdType> uva::smt::bpbd::server::lm::__G2DMapTrie::S_M_GramData< TPayloadType, TWordIdType >::TM_Gram_Id

Definition at line 79 of file g2d_map_trie.hpp.

8.65.3 Constructor & Destructor Documentation

8.65.3.1 template<typename TPayloadType , typename TWordIdType > uva::smt::bpbd::server::lm↩ ::__G2DMapTrie::S_M_GramData< TPayloadType, TWordIdType >::S_M_GramData ( ) [inline]

The basic constructor

Definition at line 91 of file g2d_map_trie.hpp.

8.65.3.2 template<typename TPayloadType , typename TWordIdType > uva::smt::bpbd::server::lm::__G2DMapTrie::S_M_GramData< TPayloadType, TWordIdType >::~S_M_GramData ( ) [inline]

The basic destructor

Definition at line 97 of file g2d_map_trie.hpp.

8.65.4 Member Function Documentation

8.65.4.1 template<typename TPayloadType , typename TWordIdType > bool uva::smt::bpbd::server::lm::__G2DMapTrie::S_M_GramData< TPayloadType, TWordIdType >::operator== (const T_Gram_Id_Key & *key*) const [inline]

The comparison operator, allows to compare two m-gram ids

Parameters

id	the m-gram id to compare with

Returns

true if the ids are equal, otherwise false

Definition at line 108 of file g2d_map_trie.hpp.

8.65.5 Member Data Documentation

8.65.5.1 template<typename TPayloadType , typename TWordIdType > TM_Gram_Id_Value_Ptr uva::smt::bpbd::server::lm::__G2DMapTrie::S_M_GramData< TPayloadType, TWordIdType >::m_id

Definition at line 84 of file g2d_map_trie.hpp.

8.65.5.2 template<typename TPayloadType , typename TWordIdType > TPayloadType uva::smt::bpbd::server::lm::__G2DMapTrie::S_M_GramData< TPayloadType, TWordIdType >::m_payload

Definition at line 86 of file g2d_map_trie.hpp.

The documentation for this struct was generated from the following file:

- inc/server/lm/models/g2d_map_trie.hpp

8.66 uva::smt::bpbd::server::decoder::sentence::sentence_decoder Class Reference

`#include <sentence_decoder.hpp>`

Public Member Functions

- sentence_decoder (const de_parameters &params, acr_bool_flag is_stop, const string &source_sent, string &target_sent)
- ~sentence_decoder ()
- void translate ()

Protected Member Functions

- prob_weight & initialize_future_costs (const size_t &start_idx, const size_t &end_idx)
- void compute_futue_costs ()
- void query_translation_model ()
- void query_reordering_model ()
- void perform_translation ()

Static Protected Member Functions

- static size_t count_words (const string &sentence)

8.66.1 Detailed Description

This class represents a sentence translator utility. It receives a sentence to translate. Performs tokenization, lowercasing, splitting it into sub-phrases, performs decoding provides recombines the result into the target sentence.

Definition at line 74 of file sentence_decoder.hpp.

8.66.2 Constructor & Destructor Documentation

8.66.2.1 uva::smt::bpbd::server::decoder::sentence::sentence_decoder::sentence_decoder (const de_parameters & *params,* acr_bool_flag *is_stop,* const string & *source_sent,* string & *target_sent*) [inline]

The basic constructor

Parameters

params	the reference to the decoder parameters
is_stop	the flag that will be set to true in case one needs to abort the translation process.
source_sent	[in] the source language sentence to translate the source sentence is expected to be tokenized, reduced, and in the lower case.
target_sent	[out] the resulting target language sentence

Definition at line 87 of file sentence_decoder.hpp.

8.66.2.2 uva::smt::bpbd::server::decoder::sentence::sentence_decoder::~sentence_decoder ( ) [inline]

The basic destructor

Definition at line 104 of file sentence_decoder.hpp.

8.66.3 Member Function Documentation

8.66.3.1 void uva::smt::bpbd::server::decoder::sentence::sentence_decoder::compute_futue_costs ( ) [inline], [protected]

Allows to compute the future costs for the sentence.

Definition at line 192 of file sentence_decoder.hpp.

8.66.3.2 static size_t uva::smt::bpbd::server::decoder::sentence::sentence_decoder::count_words (const string & *sentence*) [inline], [static], [protected]

Allows to count the number of tokens/words in the given sentence

Parameters

sentence	the sentence to count the words in

Returns

the number of words

Definition at line 250 of file sentence_decoder.hpp.

8.66.3.3 prob_weight& uva::smt::bpbd::server::decoder::sentence::sentence_decoder::initialize_future_costs (const size_t & *start_idx,* const size_t & *end_idx*) [inline], [protected]

Dynamically initialize the future costs based on the estimates from the TM and LM models.

Definition at line 146 of file sentence_decoder.hpp.

8.66.3.4 void uva::smt::bpbd::server::decoder::sentence::sentence_decoder::perform_translation ( ) [inline], [protected]

Performs the sentence translation

Definition at line 378 of file sentence_decoder.hpp.

8.66.3.5 void uva::smt::bpbd::server::decoder::sentence::sentence_decoder::query_reordering_model ( ) [inline], [protected]

Allows to query the reordering model based on the set sentence phrases

Definition at line 360 of file sentence_decoder.hpp.

8.66.3.6 void uva::smt::bpbd::server::decoder::sentence::sentence_decoder::query_translation_model ( ) [inline], [protected]

Allows to set the source sentence, this includes preparing things for decoding

Definition at line 270 of file sentence_decoder.hpp.

8.66.3.7 void uva::smt::bpbd::server::decoder::sentence::sentence_decoder::translate ( ) [inline]

This is the main method needed to be called for translating a sentence.

Definition at line 114 of file sentence_decoder.hpp.

The documentation for this class was generated from the following file:

- inc/server/decoder/sentence/sentence_decoder.hpp

8.67 uva::smt::bpbd::server::server_parameters Struct Reference

`#include <server_parameters.hpp>`

Public Member Functions

- void verify ()

Public Attributes

- string m_source_lang
- string m_target_lang
- uint16_t m_server_port
- size_t m_num_threads
- tm_parameters m_tm_params
- rm_parameters m_rm_params
- lm_parameters m_lm_params
- de_parameters m_de_params

8.67.1 Detailed Description

This structure stores the translation server parameters

Definition at line 58 of file server_parameters.hpp.

8.67.2 Member Function Documentation

8.67.2.1 void uva::smt::bpbd::server::server_parameters::verify ( ) [inline]

Allows to verify the parameters to be correct.

Definition at line 85 of file server_parameters.hpp.

8.67.3 Member Data Documentation

8.67.3.1 de_parameters uva::smt::bpbd::server::server_parameters::m_de_params

Definition at line 80 of file server_parameters.hpp.

8.67.3.2 lm_parameters uva::smt::bpbd::server::server_parameters::m_lm_params

Definition at line 77 of file server_parameters.hpp.

8.67.3.3 size_t uva::smt::bpbd::server::server_parameters::m_num_threads

Definition at line 68 of file server_parameters.hpp.

8.67.3.4 rm_parameters uva::smt::bpbd::server::server_parameters::m_rm_params

Definition at line 74 of file server_parameters.hpp.

8.67.3.5 uint16_t uva::smt::bpbd::server::server_parameters::m_server_port

Definition at line 65 of file server_parameters.hpp.

8.67.3.6 string uva::smt::bpbd::server::server_parameters::m_source_lang

Definition at line 60 of file server_parameters.hpp.

8.67.3.7 string uva::smt::bpbd::server::server_parameters::m_target_lang

Definition at line 62 of file server_parameters.hpp.

8.67.3.8 tm_parameters uva::smt::bpbd::server::server_parameters::m_tm_params

Definition at line 71 of file server_parameters.hpp.

The documentation for this struct was generated from the following file:

- inc/server/server_parameters.hpp

8.68 uva::smt::bpbd::server::decoder::stack::stack_data Struct Reference

`#include <stack_data.hpp>`

Public Member Functions

- stack_data (const de_parameters &params, acr_bool_flag is_stop, const string &source_sent, const sentence_data_map &sent_data, const rm_query_proxy &rm_query, lm_fast_query_proxy &lm_query, const add_new_state_function &add_state)

Public Attributes

- const de_parameters & m_params
- acr_bool_flag m_is_stop
- const string & m_source_sent
- const sentence_data_map & m_sent_data
- const rm_query_proxy & m_rm_query
- lm_fast_query_proxy & m_lm_query
- const add_new_state_function m_add_state

8.68.1 Detailed Description

This structure stores the shared stack-state data. This data is valid within one sentence translation and is needed by multiple states and etc

Definition at line 54 of file stack_data.hpp.

8.68.2 Constructor & Destructor Documentation

8.68.2.1 uva::smt::bpbd::server::decoder::stack::stack_data::stack_data (const de_parameters & *params,* acr_bool_flag *is_stop,* const string & *source_sent,* const sentence_data_map & *sent_data,* const rm_query_proxy & *rm_query,* lm_fast_query_proxy & *lm_query,* const add_new_state_function & *add_state*) [inline]

The basic constructor to initialize the stored references

Parameters

params	the decoder parameters
is_stop	the stopping flag
source_sent	the reference to the source sentence
sent_data	the sentence data
rm_query	the reordering model query
lm_query	the language model query to be used
add_state	the function needed to add new states

Definition at line 66 of file stack_data.hpp.

8.68.3 Member Data Documentation

8.68.3.1 const add_new_state_function uva::smt::bpbd::server::decoder::stack::stack_data::m_add_state

Definition at line 92 of file stack_data.hpp.

8.68.3.2 acr_bool_flag uva::smt::bpbd::server::decoder::stack::stack_data::m_is_stop

Definition at line 77 of file stack_data.hpp.

8.68.3.3 lm_fast_query_proxy& uva::smt::bpbd::server::decoder::stack::stack_data::m_lm_query

Definition at line 89 of file stack_data.hpp.

8.68.3.4 const de_parameters& uva::smt::bpbd::server::decoder::stack::stack_data::m_params

Definition at line 74 of file stack_data.hpp.

8.68.3.5 const rm_query_proxy& uva::smt::bpbd::server::decoder::stack::stack_data::m_rm_query

Definition at line 86 of file stack_data.hpp.

8.68.3.6 const sentence_data_map& uva::smt::bpbd::server::decoder::stack::stack_data::m_sent_data

Definition at line 83 of file stack_data.hpp.

8.68.3.7 const string& uva::smt::bpbd::server::decoder::stack::stack_data::m_source_sent

Definition at line 80 of file stack_data.hpp.

The documentation for this struct was generated from the following file:

- inc/server/decoder/stack/stack_data.hpp

8.69 uva::smt::bpbd::server::decoder::stack::stack_level Class Reference

`#include <stack_level.hpp>`

Public Member Functions

- stack_level (const de_parameters &params, acr_bool_flag is_stop)
- ~stack_level ()
- void add_state (stack_state_ptr new_state)
- void expand ()
- void get_best_trans (string &target_sent) const
- size_t get_size () const

Protected Member Functions

- bool find_recombine (stack_state_ptr &curr_state, stack_state &new_state)
- void add_last (stack_state_ptr new_state)
- void add_before (stack_state_ptr curr_state, stack_state_ptr new_state)
- void remember_best_score ()
- bool is_space_left () const
- void prune_states ()
- void insert_as_first (stack_state_ptr state)
- void insert_as_last (stack_state_ptr state)
- void insert_between (stack_state_ptr prev, stack_state_ptr next, stack_state_ptr state)
- void insert_before (stack_state_ptr curr_state, stack_state_ptr new_state)
- void remove_from_level (stack_state_ptr state)

8.69.1 Detailed Description

Represents the multi-stack level

Definition at line 45 of file stack_level.hpp.

8.69.2 Constructor & Destructor Documentation

8.69.2.1 uva::smt::bpbd::server::decoder::stack::stack_level::stack_level (const de_parameters & *params*, acr_bool_flag *is_stop*) [inline]

The basic constructor

Parameters

params	the decoder parameters, stores the reference to it
is_stop	the stop flag

Definition at line 53 of file stack_level.hpp.

8.69.2.2 uva::smt::bpbd::server::decoder::stack::stack_level::~stack_level ( ) [inline]

The basic destructor, this implementation is iterative.

Definition at line 62 of file stack_level.hpp.

8.69.3 Member Function Documentation

8.69.3.1 void uva::smt::bpbd::server::decoder::stack::stack_level::add_before (stack_state_ptr *curr_state*, stack_state_ptr *new_state*) [inline], [protected]

Allows to add a new state to the level before some existing state. The new state is to have a bigger weight that the provided current state and is to be non equal (recombinable) to any other state before. This method makes sure that

any state after the new one will be checked for a possible recombination to the new one, if yes the recombination will be done. Pruning is performed unconditionally. We known that the state satisfies the total weight threshold.

Parameters

curr_state	the pointer to the state, not NULL, we need to add the new state prior to.
new_state	the pointer to the new state, not NULL

Definition at line 252 of file stack_level.hpp.

8.69.3.2 void uva::smt::bpbd::server::decoder::stack::stack_level::add_last (stack_state_ptr *new_state*) [inline], [protected]

Allows to add the new state as the last one to the level. This new state is to have the smallest weight that all the other states in the level and is not to be equal (recombinable) to any other state to the level. Note that we know that the new state cost is within the current threshold bound.

Parameters

new_state	the new state to add as the last one, if satisfies the pruning thresholds.

Definition at line 222 of file stack_level.hpp.

8.69.3.3 void uva::smt::bpbd::server::decoder::stack::stack_level::add_state (stack_state_ptr *new_state*) [inline]

Allows to add a new state into the level

Parameters

new_state	the new state to add

Definition at line 82 of file stack_level.hpp.

8.69.3.4 void uva::smt::bpbd::server::decoder::stack::stack_level::expand ( ) [inline]

Allows to expand the stack elements, to do that this method just goes through all the stack elements one by one and expands them. We could have done this recursively but this way we avoid stack allocations so we might be just faster.

Definition at line 134 of file stack_level.hpp.

8.69.3.5 bool uva::smt::bpbd::server::decoder::stack::stack_level::find_recombine (stack_state_ptr & *curr_state*, stack_state & *new_state*) [inline], [protected]

This method allows to search for a position to insert the new state into. We known that the state satisfies the total weight threshold.

Parameters

curr_state	[out]
new_state	[in] the new state to be inserted into the list

Returns

true if the new state was recombined into an existing one, otherwise false.

Definition at line 186 of file stack_level.hpp.

8.69.3.6 void uva::smt::bpbd::server::decoder::stack::stack_level::get_best_trans (string & *target_sent*) const [inline]

Allows to get the best translation target string for this stack. To do that, it takes the first element/state in the level's ordered by costs stack and asks it to unroll itself to give its translation.

Parameters

target_sent	[out] the variable to store the translation

Definition at line 158 of file stack_level.hpp.

8.69.3.7 size_t uva::smt::bpbd::server::decoder::stack::stack_level::get_size ( ) const [inline]

This method allows to retrieve the number of stack level elements

Returns

the number of stack level elements.

Definition at line 173 of file stack_level.hpp.

8.69.3.8 void uva::smt::bpbd::server::decoder::stack::stack_level::insert_as_first (stack_state_ptr *state*) [inline], [protected]

Allows to insert the stack state as the first one in the level This method increments the level size counter. Updates the best score!

Parameters

state	the state to insert

Definition at line 359 of file stack_level.hpp.

8.69.3.9 void uva::smt::bpbd::server::decoder::stack::stack_level::insert_as_last (stack_state_ptr *state*) [inline], [protected]

Allows to insert the stack state as the last one in the level This method increments the level size counter. Updates the best score!

Parameters

state	the state to insert

Definition at line 394 of file stack_level.hpp.

8.69.3.10 void uva::smt::bpbd::server::decoder::stack::stack_level::insert_before (stack_state_ptr *curr_state,* stack_state_ptr *new_state*) [inline], [protected]

Allows to insert a new element before the given stack element in the level list This method increments the level size counter. Updates the best score!

Parameters

curr_state	the state before which the new state is to be inserted, not NULL
new_state	the state to be inserted, NOT NULL

Definition at line 460 of file stack_level.hpp.

8.69.3.11 void uva::smt::bpbd::server::decoder::stack::stack_level::insert_between (stack_state_ptr *prev,* stack_state_ptr *next,* stack_state_ptr *state*) [inline], [protected]

Allows to insert the stack state in between the given two elements Note that the next and previous states are to be different! This method increments the level size counter.

Parameters

prev	the pointer reference to the prev state, NOT NULL
next	the pointer reference to the next state, NOT NULL
state	the state to insert, NUL NULL

Definition at line 428 of file stack_level.hpp.

8.69.3.12 bool uva::smt::bpbd::server::decoder::stack::stack_level::is_space_left ( ) const [inline], [protected]

Allows to check if there is still space left for adding states into the level If there is no space left then we can still add states but we shall do histogram pruning afterwards in order to keep the stack size within the capacity limits.

Returns

true if there is empty space left for adding states

Definition at line 315 of file stack_level.hpp.

8.69.3.13 void uva::smt::bpbd::server::decoder::stack::stack_level::prune_states ( ) [inline], [protected]

This method makes sure there is not too many elements in the stack, the last ones are removed. This method decrements the level size counter.

Definition at line 326 of file stack_level.hpp.

8.69.3.14 void uva::smt::bpbd::server::decoder::stack::stack_level::remember_best_score ( ) [inline], [protected]

Allows to update the best score, or rather threshold for threshold pruning.

Definition at line 297 of file stack_level.hpp.

8.69.3.15 void uva::smt::bpbd::server::decoder::stack::stack_level::remove_from_level (stack_state_ptr *state*) [inline], [protected]

Allows to remove the given state from the level. The state is not destroyed, its memory is not deleted. This method decrements the level size counter. The given state must be within the level list of states!

Parameters

state	the state to be destroyed

Definition at line 480 of file stack_level.hpp.

The documentation for this class was generated from the following file:

- inc/server/decoder/stack/stack_level.hpp

8.70 uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH > Class Template Reference

`#include <stack_data.hpp>`

Public Types

- typedef state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRA↩ M_QUERY_LENGTH > state_data

Public Member Functions

- stack_state_templ (const stack_data &data)
- stack_state_templ (stack_state_ptr parent)
- stack_state_templ (stack_state_ptr parent, const int32_t begin_pos, const int32_t end_pos, const typename state_data::covered_info &covered, tm_const_target_entry *target)
- ~stack_state_templ ()
- uint32_t get_stack_level () const
- void expand ()
- void get_translation (string &target_sent) const
- bool operator< (const stack_state &other) const
- bool operator== (const stack_state &other) const
- bool operator!= (const stack_state &other) const
- bool is_above_threshold (const prob_weight &score_bound) const
- void recombine_from (stack_state_ptr other_state)

Protected Member Functions

- void cut_the_tail (stack_state_ptr tail)
- void count_and_prune (size_t state_count, stack_state_ptr tail)
- void merge_recomb_from (const stack_state_ptr recomb_from, const size_t recomb_from_count)
- void expand_length_if_not_covered (int32_t &curr_pos, size_t &num_exp)
- void expand_left ()
- void expand_right ()
- void expand_length (const size_t start_pos)
- template<bool single_word>
  void expand_trans (const size_t start_pos, const size_t end_pos)

Friends

- class stack_level

8.70.1 Detailed Description

template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGT↩ H>class uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENG↩ TH, MAX_M_GRAM_QUERY_LENGTH >

This is the translation stack state class that is responsible for the sentence translation

Definition at line 37 of file stack_data.hpp.

8.70.2 Member Typedef Documentation

8.70.2.1 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_Q↩ UERY_LENGTH> typedef state_data_templ<NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH> uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::state_data

Definition at line 69 of file stack_state.hpp.

8.70.3 Constructor & Destructor Documentation

8.70.3.1 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::stack_state_templ (const stack_data & *data*) [inline]

The basic constructor for the begin stack state

Parameters

data	the shared data container

Definition at line 75 of file stack_state.hpp.

8.70.3.2 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::stack_state_templ (stack_state_ptr *parent*) [inline]

The basic constructor for the end stack state

Parameters

parent	the parent state pointer, NOT NULL!
prev_history	the previous translation history

Definition at line 86 of file stack_state.hpp.

8.70.3.3 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::stack_state_templ (stack_state_ptr *parent*, const int32_t *begin_pos*, const int32_t *end_pos*, const typename state_data::covered_info & *covered*, tm_const_target_entry * *target*) [inline]

The basic constructor for the non-begin/end stack state

Parameters

parent	the parent state pointer, NOT NULL!
begin_pos	this state translated source phrase begin position
end_pos	this state translated source phrase end position
covered	the pre-cooked covered vector, for efficiency reasons.
target	the new translation target

Definition at line 100 of file stack_state.hpp.

8.70.3.4 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::~stack_state_templ ( ) [inline]

The basic destructor, should free all the allocated resources. Deletes the states that are recombined into this state as they are not in any stack level

Definition at line 116 of file stack_state.hpp.

8.70.4 Member Function Documentation

8.70.4.1 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> void uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::count_and_prune (size_t *state_count,* stack_state_ptr *tail*) [inline], [protected]

Count the number of states in the remaining tail, once the maximum capacity is reached the remaining tail elements are to be deleted.

Parameters

state_count	the number of elements up until the tail element
tail	the pointer to the firt tail element

Definition at line 391 of file stack_state.hpp.

8.70.4.2 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> void uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::cut_the_tail (stack_state_ptr *tail*) [inline], [protected]

Allows to cut the tail of states starting from this one. The states present in the cut tail are to be deleted.

Parameters

tail	the tails of staits to delete

Definition at line 362 of file stack_state.hpp.

8.70.4.3 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> void uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::expand ( ) [inline]

Allows the state to expand itself, it will add itself to the proper stack.

Definition at line 154 of file stack_state.hpp.

8.70.4.4 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> void uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::expand_left ( ) [inline], [protected]

Expand to the left of the last phrase, for all the possible of start positions

Definition at line 537 of file stack_state.hpp.

8.70.4.5 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> void uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::expand_length (const size_t *start_pos*) [inline], [protected]

Allows to expand for all the possible phrase lengths

Definition at line 636 of file stack_state.hpp.

8.70.4.6 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> void uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::expand_length_if_not_covered (int32_t & *curr_pos,* size_t & *num_exp*) [inline], [protected]

Allows to expand the lengths if not the word given by the current position is not covered.

Parameters

curr_pos	the reference to the current position, will be decremented by the method by one
num_exp	the reference to the number of positions we could expand from will be incremented by this method by one if an expansion is possible.

Definition at line 518 of file stack_state.hpp.

8.70.4.7 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> void uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::expand_right ( ) [inline], [protected]

Expand to the right of the last phrase, for all the possible of start positions

Definition at line 593 of file stack_state.hpp.

8.70.4.8 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> template<bool single_word> void uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::expand_trans (const size_t *start_pos,* const size_t *end_pos*) [inline], [protected]

Allows to expand for all the possible translations

Definition at line 668 of file stack_state.hpp.

8.70.4.9 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> uint32_t uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::get_stack_level ( ) const [inline]

Allows to get the stack level, the latter is equal to the number of so far translated words.

Returns

the stack level

Definition at line 145 of file stack_state.hpp.

8.70.4.10 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> void uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::get_translation (string & *target_sent*) const [inline]

Allows to get the translation ending in this state.

Parameters

target_sent	[out] the variable to store the translation

Definition at line 182 of file stack_state.hpp.

8.70.4.11 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> bool uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::is_above_threshold (const prob_weight & *score_bound*) const [inline]

Allows to check if the given new state is within the threshold limit.

Parameters

score_bound	the bound to compare with

Returns

true if the state's totl cost is >= score_bound, otherwise false

Definition at line 289 of file stack_state.hpp.

8.70.4.12 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> void uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::merge_recomb_from (const stack_state_ptr *recomb_from,* const size_t *recomb_from_count*) [inline], [protected]

Allows to combine the two recombine from lists together. We combine them into the current list. The states that are remained over are deleted. There remaining states are the ones from both lists that go outside the list capacity. This method must only be called if the m_recomb_from != NULL or alternatively m_recomb_from_count > 0.

Parameters

recomb_from	the recombine from list with at least one element
recomb_from_count	the number of elements in the recomb from list

Definition at line 415 of file stack_state.hpp.

8.70.4.13 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> bool uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::operator!= (const stack_state & *other*) const [inline]

Allows to compare two states for not being equal, this is an inverse of the == operator.

Parameters

other	the other state to compare with

Returns

true if this state is not equal to the other one, otherwise false.

Definition at line 279 of file stack_state.hpp.

8.70.4.14 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> bool uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::operator< (const stack_state & *other*) const [inline]

Allows to compare two states, the comparison is based on the state total score. The state with the bigger total score is considered to be bigger, i.e. more probable. The state with the smalle total score is considered to be smaller, i.e. less probable.

Parameters

other	the other state to compare with

Returns

true if this state is smaller than the other one

Definition at line 223 of file stack_state.hpp.

8.70.4.15 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> bool uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::operator== (const stack_state & *other*) const [inline]

Allows to compare two states, the states are equal if the solve the same sub-problem i.e. are eligible for recombination. The states are equal if and only if:

1. They have the same last translated word
2. They have the same history of target words
3. They cover the same source words

Parameters

other	the other state to compare with

Returns

true if this state is equal to the other one, otherwise false.

Definition at line 251 of file stack_state.hpp.

8.70.4.16 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> void uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::recombine_from (stack_state_ptr *other_state*) [inline]

Allows to add a state recombined into this one, i.e. the one equivalent to this one but having the lower value of the total cost. In case this state already has too many states recombined into this one and the new state probability is lower than that of the others, then we just delete it. Also, if there were states recombined into the other one, then they have lower costs, so proper merging of them is to be done as well. Eventually the states recombined into this one must have their m_recomb_from arrays empty.

Parameters

new_state	the state to recombine into this one.

Definition at line 308 of file stack_state.hpp.

8.70.5 Friends And Related Function Documentation

8.70.5.1 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> friend class stack_level [friend]

Definition at line 721 of file stack_state.hpp.

The documentation for this class was generated from the following files:

- inc/server/decoder/stack/stack_data.hpp
- inc/server/decoder/stack/stack_state.hpp

8.71 uva::utils::monitor::stat_monitor Class Reference

`#include <statistics_monitor.hpp>`

Static Public Member Functions

- static void get_mem_stat (TMemotyUsage &memStat)
- static double get_cpu_time ()

8.71.1 Detailed Description

This class is responsible for monitoring the program statistics, such as the used memory and CPU times. This class is a trivial singleton

Definition at line 66 of file statistics_monitor.hpp.

8.71.2 Member Function Documentation

8.71.2.1 double uva::utils::monitor::stat_monitor::get_cpu_time ( ) [static]

This function returns the current CPU time as given in the article http://nadeausoftware.↩com/articles/2012/03/c_c_tip_how_measure_cpu_time_benchmarking

Returns

Returns the amount of CPU time used by the current process, in seconds, or -1.0 if an error occurred.

Definition at line 144 of file statistics_monitor.cpp.

8.71.2.2 void uva::utils::monitor::stat_monitor::get_mem_stat (TMemotyUsage & *memStat*) [static]

Allows to get the current memory usage of the process.

Parameters

memStat	this is an out parameter that will store the obtained data

Exceptions

Exception	in case the memory statistics can not be obtained.

This implementation is derived from http://locklessinc.com/articles/memory_usage/ This here is actually C-style code and also pretty ugly.

Definition at line 67 of file statistics_monitor.cpp.

The documentation for this class was generated from the following files:

- inc/common/utils/monitor/statistics_monitor.hpp
- src/common/utils/monitor/statistics_monitor.cpp

8.72 uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_↩SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH > Struct Template Reference

`#include <state_data.hpp>`

Public Types

- typedef circular_queue< word_uid, MAX_M_GRAM_QUERY_LENGTH > state_frame
- typedef bitset< NUM_WORDS_PER_SENTENCE > covered_info

Public Member Functions

- state_data_templ (const stack_data &stack_data)
- state_data_templ (const state_data_templ &prev_state_data)
- state_data_templ (const state_data_templ &prev_state_data, const int32_t &begin_pos, const int32_t &end_pos, const covered_info &covered, tm_const_target_entry *target)
- string covered_to_string () const

Public Attributes

- const stack_data & m_stack_data
- const int32_t m_s_begin_word_idx
- const int32_t m_s_end_word_idx
- const phrase_length m_stack_level
- tm_const_target_entry *const m_target
- const rm_entry & rm_entry_data
- const state_frame m_trans_frame
- phrase_length m_begin_lm_level
- const covered_info m_covered
- const prob_weight m_partial_score
- const prob_weight m_total_score

Static Public Attributes

- static constexpr int32_t UNDEFINED_WORD_IDX = -1
- static constexpr int32_t ZERRO_WORD_IDX = UNDEFINED_WORD_IDX + 1

8.72.1 Detailed Description

template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH>struct uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >

This structure is needed to store the common state data that however changes/mutates from state to state and thus is to be passed on from each state to its child.

Definition at line 65 of file state_data.hpp.

8.72.2 Member Typedef Documentation

8.72.2.1 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> typedef bitset<NUM_WORDS_PER_SENTENCE> uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::covered_info

Definition at line 70 of file state_data.hpp.

8.72.2.2 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> typedef circular_queue<word_uid, MAX_M_GRAM_QUERY_LENGTH > uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::state_frame

Definition at line 67 of file state_data.hpp.

8.72.3 Constructor & Destructor Documentation

8.72.3.1 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::state_data_templ (const stack_data & *stack_data*) [inline]

The basic constructor that is to be used for the BEGIN STATE

Parameters

stack_data	the general shared stack data reference
is_begin_end	this flag allows to detect whether this data is created for the begin or end tag. If true then it is for the begin tag, if false then it is for the end tag

Definition at line 83 of file state_data.hpp.

8.72.3.2 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::state_data_templ (const state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH > & *prev_state_data*) [inline]

The basic constructor that is to be used for the END STATE

Parameters

stack_data	the general shared stack data reference
is_begin_end	this flag allows to detect whether this data is created for the begin or end tag. If true then it is for the begin tag, if false then it is for the end tag

Definition at line 108 of file state_data.hpp.

8.72.3.3 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::state_data_templ (const state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH > & *prev_state_data,* const int32_t & *begin_pos,* const int32_t & *end_pos,* const covered_info & *covered,* tm_const_target_entry * *target*) [inline]

The basic constructor that is to be used for an INTERMEDIATE STATE data, it takes the parent state data and the new data to be stored/merged with the parent's data.

Parameters

state_data	the constant reference to the parent state data
begin_pos	this state translated source phrase begin position
end_pos	this state translated source phrase end position

target	the pointer to the target translation of the source phrase

Definition at line 142 of file state_data.hpp.

8.72.4 Member Function Documentation

8.72.4.1 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> string uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::covered_to_string ( ) const [inline]

Allows to give the string representation of the covered vector

Returns

the string representation of the covered vector

Definition at line 172 of file state_data.hpp.

8.72.5 Member Data Documentation

8.72.5.1 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> phrase_length uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::m_begin_lm_level

Definition at line 212 of file state_data.hpp.

8.72.5.2 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> const covered_info uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::m_covered

Definition at line 215 of file state_data.hpp.

8.72.5.3 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> const prob_weight uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::m_partial_score

Definition at line 218 of file state_data.hpp.

8.72.5.4 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> const int32_t uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::m_s_begin_word_idx

Definition at line 192 of file state_data.hpp.

8.72.5.5 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> const int32_t uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::m_s_end_word_idx

Definition at line 194 of file state_data.hpp.

8.72.5.6 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> const stack_data& uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::m_stack_data

Definition at line 189 of file state_data.hpp.

8.72.5.7 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> const phrase_length uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::m_stack_level

Definition at line 197 of file state_data.hpp.

8.72.5.8 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> tm_const_target_entry* const uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::m_target

Definition at line 200 of file state_data.hpp.

8.72.5.9 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> const prob_weight uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::m_total_score

Definition at line 223 of file state_data.hpp.

8.72.5.10 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> const state_frame uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::m_trans_frame

Definition at line 209 of file state_data.hpp.

8.72.5.11 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> const rm_entry& uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::rm_entry_data

Definition at line 203 of file state_data.hpp.

8.72.5.12 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> constexpr int32_t uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::UNDEFINED_WORD_IDX = -1 [static]

Definition at line 73 of file state_data.hpp.

8.72.5.13 template<size_t NUM_WORDS_PER_SENTENCE, size_t MAX_HISTORY_LENGTH, size_t MAX_M_GRAM_QUERY_LENGTH> constexpr int32_t uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >::ZERRO_WORD_IDX = UNDEFINED_WORD_IDX + 1 [static]

Definition at line 74 of file state_data.hpp.

The documentation for this struct was generated from the following file:

- inc/server/decoder/stack/state_data.hpp

8.73 uva::smt::bpbd::server::lm::m_grams::m_gram_id::T_Gram_Id_Key Struct Reference

`#include <m_gram_id.hpp>`

Public Attributes

- TM_Gram_Id_Value_Ptr m_id
- uint8_t m_len_bytes

8.73.1 Detailed Description

This structure defined the m-gram id key which consists of the m-gram id and its length in bytes

Definition at line 65 of file m_gram_id.hpp.

8.73.2 Member Data Documentation

8.73.2.1 TM_Gram_Id_Value_Ptr uva::smt::bpbd::server::lm::m_grams::m_gram_id::T_Gram_Id_Key::m_id

Definition at line 66 of file m_gram_id.hpp.

8.73.2.2 uint8_t uva::smt::bpbd::server::lm::m_grams::m_gram_id::T_Gram_Id_Key::m_len_bytes

Definition at line 67 of file m_gram_id.hpp.

The documentation for this struct was generated from the following file:

- inc/server/lm/mgrams/m_gram_id.hpp

8.74 uva::utils::containers::utils::T_IS_COMPARE_FUNC< ELEM_TYPE > Struct Template Reference

`#include <array_utils.hpp>`

Public Types

- typedef std::function< bool(const ELEM_TYPE &, const ELEM_TYPE &) > func_type
- typedef bool(* func_ptr) (const ELEM_TYPE &, const ELEM_TYPE &)

8.74.1 Detailed Description

template<typename ELEM_TYPE>struct uva::utils::containers::utils::T_IS_COMPARE_FUNC< ELEM_TYPE >

Define the function type for the comparison function

Definition at line 50 of file array_utils.hpp.

8.74.2 Member Typedef Documentation

8.74.2.1 template<typename ELEM_TYPE> typedef bool(* uva::utils::containers::utils::T_IS_COMPARE_FUNC< ELEM_TYPE >::func_ptr) (const ELEM_TYPE &, const ELEM_TYPE &)

Definition at line 52 of file array_utils.hpp.

8.74.2.2 template<typename ELEM_TYPE> typedef std::function<bool(const ELEM_TYPE &, const ELEM_TYPE &) > uva::utils::containers::utils::T_IS_COMPARE_FUNC< ELEM_TYPE >::func_type

Definition at line 51 of file array_utils.hpp.

The documentation for this struct was generated from the following file:

- inc/common/utils/containers/array_utils.hpp

8.75 uva::smt::bpbd::server::lm::arpa::TAddGramFunct< WordIndexType > Struct Template Reference

`#include <lm_gram_builder.hpp>`

Public Types

- typedef std::function< void(const model_m_gram &) > func

8.75.1 Detailed Description

template<typename WordIndexType>struct uva::smt::bpbd::server::lm::arpa::TAddGramFunct< WordIndexType >

Definition at line 50 of file lm_gram_builder.hpp.

8.75.2 Member Typedef Documentation

8.75.2.1 template<typename WordIndexType> typedef std::function<void (const model_m_gram&) > uva::smt::bpbd::server::lm::arpa::TAddGramFunct< WordIndexType >::func

Definition at line 51 of file lm_gram_builder.hpp.

The documentation for this struct was generated from the following file:

- inc/server/lm/builders/lm_gram_builder.hpp

8.76 uva::smt::bpbd::server::lm::__C2WArrayTrie::TCtxIdProbData Struct Reference

`#include <c2w_array_trie.hpp>`

Public Attributes

- TShortId word_id
- TShortId ctx_id
- prob_weight prob

8.76.1 Detailed Description

Stores the information about the context id, word id and corresponding probability This data structure is to be used for the N-Gram data, as there are no back-offs It is used to store the N-gram data for the last Trie level N.

Parameters

ctx_id	the context id
word_id	the word id
prob	the probability data

Definition at line 81 of file c2w_array_trie.hpp.

8.76.2 Member Data Documentation

8.76.2.1 TShortId uva::smt::bpbd::server::lm::__C2WArrayTrie::TCtxIdProbData::ctx_id

Definition at line 83 of file c2w_array_trie.hpp.

8.76.2.2 prob_weight uva::smt::bpbd::server::lm::__C2WArrayTrie::TCtxIdProbData::prob

Definition at line 84 of file c2w_array_trie.hpp.

8.76.2.3 TShortId uva::smt::bpbd::server::lm::__C2WArrayTrie::TCtxIdProbData::word_id

Definition at line 82 of file c2w_array_trie.hpp.

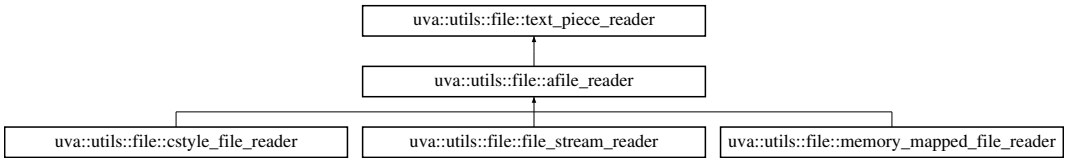
The documentation for this struct was generated from the following file:

- inc/server/lm/models/c2w_array_trie.hpp

8.77 uva::utils::file::text_piece_reader Class Reference

`#include <text_piece_reader.hpp>`

Inheritance diagram for uva::utils::file::text_piece_reader:



Public Member Functions

- text_piece_reader ()
- text_piece_reader (const void *begin_ptr, const size_t len)
- text_piece_reader (const text_piece_reader &other)
- void set (const void *begin_ptr, const size_t len)
- const char * get_begin_c_str () const
- const char * get_rest_c_str () const
- string get_rest_str () const
- const void * get_begin_ptr () const
- size_t length () const

- template<const size_t LEN_LIMIT>
  void copy_string (const text_piece_reader &other)
- template<const char delim, const uint8_t delim_len = 1>
  bool get_first (text_piece_reader &out)
- template<const char delim, const uint8_t delim_card = 1>
  bool get_last (text_piece_reader &out)
- bool has_more ()
- bool get_first_line (text_piece_reader &out)
- bool get_first_space (text_piece_reader &out)
- bool get_last_space (text_piece_reader &out)
- bool get_first_tab (text_piece_reader &out)
- char operator[] (size_t idx)
- bool operator== (const text_piece_reader &other) const
- bool operator!= (const text_piece_reader &other) const
- bool operator== (const char *other) const
- bool operator!= (const char *other) const
- bool operator== (const string &other) const
- bool operator!= (const string &other) const
- const string & str () const

Protected Member Functions

- template<const char delim, const uint8_t delim_len>
  const char * find_first_subseq ()

8.77.1 Detailed Description

This basic storage class that stores the pointer to pre-allocated memory plus the length of this piece. This class can be used to represent a piece of text, a line or a word or any arbitrary piece of memory. Note that, the string here is not necessarily \0 terminated and the text memory can be Gb large! Also the memory is not managed by the class.

Definition at line 61 of file text_piece_reader.hpp.

8.77.2 Constructor & Destructor Documentation

8.77.2.1 uva::utils::file::text_piece_reader::text_piece_reader ( ) [inline]

The basic constructor initializes empty text

Definition at line 84 of file text_piece_reader.hpp.

8.77.2.2 uva::utils::file::text_piece_reader::text_piece_reader (const void * *begin_ptr*, const size_t *len*) [inline], [explicit]

The constructor.

Parameters

beginPtr	the pointer to the begin of the text
len	the length of the text

Definition at line 93 of file text_piece_reader.hpp.

8.77.2.3 uva::utils::file::text_piece_reader::text_piece_reader (const text_piece_reader & *other*) [inline]

The copy constructor.

Parameters

other	the const reference to the object to copy from

Definition at line 102 of file text_piece_reader.hpp.

8.77.3 Member Function Documentation

8.77.3.1 template<const size_t LEN_LIMIT> void uva::utils::file::text_piece_reader::copy_string (const text_piece_reader & *other*) [inline]

This method allows to copy the string of one text piece into another. The copying process re-sets the internal cursor and remaining length to read.

Parameters

other	the element to copy from
limit	the maximum length allowed to be copied from the source (other) if the source length is larger - an exception will be raised!

Definition at line 185 of file text_piece_reader.hpp.

8.77.3.2 template<const char delim, const uint8_t delim_len> const char* uva::utils::file::text_piece_reader::find_first_subseq ( ) [inline],[protected]

Allows to find a sub-sequence of characters in the forward manner

Returns

the pointer to the last character in the subsequence or NULL if nothing is found

Definition at line 512 of file text_piece_reader.hpp.

8.77.3.3 const char* uva::utils::file::text_piece_reader::get_begin_c_str ( ) const [inline]

Allows to get the pointer to the beginning of the text This is a C string that is returned BUT there is no \0 termination and it can be Gb long!

Returns

the pointer to the beginning of the text

Definition at line 137 of file text_piece_reader.hpp.

8.77.3.4 const void* uva::utils::file::text_piece_reader::get_begin_ptr ( ) const [inline]

Allows to get the pointer to the beginning of the text

Returns

the pointer to the beginning of the text

Definition at line 165 of file text_piece_reader.hpp.

8.77.3.5 template<const char delim, const uint8_t delim_len = 1> bool uva::utils::file::text_piece_reader::get_first (text_piece_reader & *out*) [inline]

This function searches forward for the first occurrence of the argument delimiter symbol.

Parameters

delim	the delimiter we are looking for
delim_len	the number of times in a row the delimiter shall occur, default is 1
out	the out parameter - the substring until the first next found delimiter or the entire string if the delimiter was not found

Returns

true if a text piece was read, otherwise false (end of file)

Definition at line 211 of file text_piece_reader.hpp.

8.77.3.6 bool uva::utils::file::text_piece_reader::get_first_line (text_piece_reader & *out*) [inline]

This function, searches forward for the first end of line char or until the end of the text and then sets the data about the found region into the provided output parameter.

Parameters

out	the out parameter - the read line

Returns

true if data was read, otherwise false

Definition at line 353 of file text_piece_reader.hpp.

8.77.3.7 bool uva::utils::file::text_piece_reader::get_first_space (text_piece_reader & *out*) [inline]

This function, searches forward for the first space char or until the end of the text and then sets the data about the found region into the provided output parameter.

Parameters

out	the out parameter - the read line

Returns

true if data was read, otherwise false

Definition at line 366 of file text_piece_reader.hpp.

8.77.3.8 bool uva::utils::file::text_piece_reader::get_first_tab (text_piece_reader & *out*) [inline]

This function, searches forward for the first end of tab char or until the end of the text and then sets the data about the found region into the provided output parameter.

Parameters

out	the out parameter - the read line

Returns

true if data was read, otherwise false

Definition at line 393 of file text_piece_reader.hpp.

8.77.3.9 template<const char delim, const uint8_t delim_card = 1> bool uva::utils::file::text_piece_reader::get_last (text_piece_reader & *out*) [inline]

This function searches backwards for the first occurrence of the argument delimiter symbol.

Parameters

delim	the delimiter we are looking for
delim_card	the number of times in a row the delimiter shall occur, default is 1
out	the out parameter - the substring from the first next found delimiter till the end of the string or the entire string if the delimiter was not found

Returns

true if a line was read, otherwise false (end of file)

Definition at line 287 of file text_piece_reader.hpp.

8.77.3.10 bool uva::utils::file::text_piece_reader::get_last_space (text_piece_reader & *out*) [inline]

This function, from the end position, searches for the space char or until the beginning of the text and then sets the data about the found region into the provided output parameter. I.e. we get a sub-string: [begin_index, last_↩ space_idx)

Parameters

out	the out parameter - the read line

Returns

true if data was read, otherwise false

Definition at line 380 of file text_piece_reader.hpp.

8.77.3.11 const char* uva::utils::file::text_piece_reader::get_rest_c_str ( ) const [inline]

Allows to get the pointer to the remainder of the text This is a C string that is returned BUT there is no \0 termination and it can be Gb long!

Returns

the pointer to the remainder of the text

Definition at line 147 of file text_piece_reader.hpp.

8.77.3.12 string uva::utils::file::text_piece_reader::get_rest_str ( ) const [inline]

Allows to get the pointer to the remainder of the text This is a C string that is returned BUT there is no \0 termination and it can be Gb long!

Returns

the pointer to the remainder of the text

Definition at line 157 of file text_piece_reader.hpp.

8.77.3.13 bool uva::utils::file::text_piece_reader::has_more ( ) [inline]

Allows to check if there is something left to read

Returns

true if there is yet something to read, otherwise false

Definition at line 342 of file text_piece_reader.hpp.

8.77.3.14 size_t uva::utils::file::text_piece_reader::length ( ) const [inline]

Allows to get the length of the text

Returns

the length of the text

Definition at line 173 of file text_piece_reader.hpp.

8.77.3.15 bool uva::utils::file::text_piece_reader::operator!= (const text_piece_reader & *other*) const [inline]

The comparison operator implementation

Parameters

other	text piece to compare with

Definition at line 434 of file text_piece_reader.hpp.

8.77.3.16 bool uva::utils::file::text_piece_reader::operator!= (const char * *other*) const [inline]

The comparison operator implementation

Parameters

other	a c_string to compare with

Definition at line 457 of file text_piece_reader.hpp.

8.77.3.17 bool uva::utils::file::text_piece_reader::operator!= (const string & *other*) const [inline]

The comparison operator implementation

Parameters

other	a c_string to compare with

Definition at line 475 of file text_piece_reader.hpp.

8.77.3.18 bool uva::utils::file::text_piece_reader::operator== (const text_piece_reader & *other*) const [inline]

The comparison operator implementation

Parameters

other	text piece to compare with

Definition at line 421 of file text_piece_reader.hpp.

8.77.3.19 bool uva::utils::file::text_piece_reader::operator== (const char * *other*) const [inline]

The comparison operator implementation

Parameters

other	a c_string to compare with

Definition at line 443 of file text_piece_reader.hpp.

8.77.3.20 bool uva::utils::file::text_piece_reader::operator== (const string & *other*) const [inline]

The comparison operator implementation

Parameters

other	a c_string to compare with

Definition at line 466 of file text_piece_reader.hpp.

8.77.3.21 char uva::utils::file::text_piece_reader::operator[ ] (size_t *idx*) [inline]

Allows to get the character at the given index, if the index stays within the text length bounds.

Parameters

idx	the character index

Returns

Definition at line 405 of file text_piece_reader.hpp.

8.77.3.22 void uva::utils::file::text_piece_reader::set (const void * *begin_ptr*, const size_t *len*) [inline]

Allows to set the text

Parameters

beginPtr	the pointer to the beginning of the text
len	the length of the text

Definition at line 116 of file text_piece_reader.hpp.

8.77.3.23 const string& uva::utils::file::text_piece_reader::str ( ) const [inline]

Allows to convert the line to string object

Returns

the resulting line

Definition at line 484 of file text_piece_reader.hpp.

The documentation for this class was generated from the following file:

- inc/common/utils/file/text_piece_reader.hpp

8.78 uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type > Class Template Reference

`#include <tm_basic_builder.hpp>`

Public Member Functions

- tm_basic_builder (const tm_parameters &params, model_type &model, reader_type &reader)
- ~tm_basic_builder ()
- void build ()

Protected Member Functions

- float post_process_feature (const float feature, const float lambda)
- template<bool is_get_weights>
  bool process_features (text_piece_reader weights, size_t &num_features, prob_weight *storage)
- bool is_good_features (text_piece_reader rest, size_t &tmp_features_size, prob_weight *tmp_features)
- void process_target_entry (tm_source_entry *source_entry, text_piece_reader &rest, size_t &count_ref, size_t &tmp_features_size, prob_weight *tmp_features)
- template<bool count_or_build>
  void parse_tm_file ()
- void count_source_phrases ()
- void process_source_entries ()
- void add_unk_translation ()

8.78.1 Detailed Description

template<typename model_type, typename reader_type>class uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type >

This class represents a basic reader of the translation model. It allows to read a text-formatted translation model and to put it into the given instance of the model class. It assumes the simple text model format as used by Oyster or Moses. See http://www.statmt.org/moses/?n=Moses.Tutorial for some info. The translation model is also commonly known as a phrase table.

Definition at line 84 of file tm_basic_builder.hpp.

8.78.2 Constructor & Destructor Documentation

8.78.2.1 template<typename model_type , typename reader_type > uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type >::tm_basic_builder (const tm_parameters & *params,* model_type & *model,* reader_type & *reader*) [inline]

The basic constructor of the builder object params the model parameters

Parameters

model	the model to put the data into
reader	the reader to read the data from

Definition at line 93 of file tm_basic_builder.hpp.

8.78.2.2 template<typename model_type , typename reader_type > uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type >::~tm_basic_builder ( ) [inline]

The basic destructor

Definition at line 103 of file tm_basic_builder.hpp.

8.78.3 Member Function Documentation

8.78.3.1 template<typename model_type , typename reader_type > void uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type >::add_unk_translation ( ) [inline], [protected]

Allows to add an unk entry to the model

Definition at line 447 of file tm_basic_builder.hpp.

8.78.3.2 template<typename model_type , typename reader_type > void uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type >::build ( ) [inline]

Allows to build the model by reading from the reader object. This is a two step process as first we need the number of distinct source phrases.

Definition at line 113 of file tm_basic_builder.hpp.

8.78.3.3 template<typename model_type , typename reader_type > void uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type >::count_source_phrases ( ) [inline], [protected]

Allows to count and set the number of source phrases

Definition at line 413 of file tm_basic_builder.hpp.

8.78.3.4 template<typename model_type , typename reader_type > bool uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type >::is_good_features (text_piece_reader *rest,* size_t & *tmp_features_size,* prob_weight * *tmp_features*) [inline], [protected]

Allows to check if whether the probability weights satisfy the filtering thresholds.

Parameters

rest	the part of the source entry containing the target and the weights
tmp_features	the temporary weights storage

Returns

true if the conditions are satisfied, otherwise false

Definition at line 207 of file tm_basic_builder.hpp.

8.78.3.5 template<typename model_type , typename reader_type > template<bool count_or_build> void uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type >::parse_tm_file ( ) [inline], [protected]

Allows to parse the TM model file and do two things depending on the value of the template parameter:

1. Count the number of valid entries
2. Build the TM model NOTE: This two pass parsing is not optimal but we have to do it as we need to know the number of valid entries beforehand, an optimization might be needed!

Parameters

count_or_build	if true then count if false then build

Definition at line 291 of file tm_basic_builder.hpp.

8.78.3.6 template<typename model_type , typename reader_type > float uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type >::post_process_feature (const float *feature,* const float *lambda*) [inline], [protected]

Allows to post-process a single feature, i.e. do: log10(feature)∗lambda

Parameters

feature	the feature to post-process
lambda	the lambda weight to multiply the log10 feature with

Returns

the post-processed feature

Definition at line 135 of file tm_basic_builder.hpp.

8.78.3.7 template<typename model_type , typename reader_type > template<bool is_get_weights> bool uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type >::process_features (text_piece_reader *weights,* size_t & *num_features,* prob_weight ∗ *storage*) [inline], [protected]

Allows to extract the features from the text piece and to check that they are valid with respect to the option bound If needed the weights will be converted to log scale and multiplied with the lambda factors

Parameters

is_get_weights	if the weights are to be retrieved or just checked
weights	[in] the text piece with weights, that starts with a space!
num_features	[out] the number of read features if they satisfy on the constraints
storage	[out] the read and post-processed features features if they satisfy on the constraints

Returns

true if the features satisfy the constraints, otherwise false

Definition at line 154 of file tm_basic_builder.hpp.

8.78.3.8 template<typename model_type , typename reader_type > void uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type >::process_source_entries ( ) [inline], [protected]

Allows to process translations.

Definition at line 434 of file tm_basic_builder.hpp.

8.78.3.9 template<typename model_type , typename reader_type > void uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type >::process_target_entry (tm_source_entry ∗ *source_entry,* text_piece_reader & *rest,* size_t & *count_ref,* size_t & *tmp_features_size,* prob_weight ∗ *tmp_features*) [inline], [protected]

The line format assumes source to target and then at least four weights as given by: http://www.statmt.org/moses/?n=FactoredTraining.ScorePhrases Currently, four different phrase translation scores

are computed: inverse phrase translation probability phi(f|e) inverse lexical weighting lex(f|e) direct phrase translation probability phi(e|f) direct lexical weighting lex(e|f) Previously, there was another score: phrase penalty (always exp(1) = 2.718) The latter is considered optional, all the other elements followed on the translation line are now skipped.

Parameters

source_entry	the pointer to the source entry for which this translation is
rest	stores the line to be parsed into a translation entry
count_ref	[in/out] the number of remaining entries
tmp_features_↩ size	[out] the number of read features
tmp_features	the temporary feature storage

Definition at line 235 of file tm_basic_builder.hpp.

The documentation for this class was generated from the following file:

- inc/server/tm/builders/tm_basic_builder.hpp

8.79 uva::smt::bpbd::server::tm::models::tm_basic_model Class Reference

`#include <tm_basic_model.hpp>`

Public Types

- typedef fixed_size_hashmap< tm_source_entry, const phrase_uid & > tm_source_entry_map

Public Member Functions

- tm_basic_model ()
- ~tm_basic_model ()
- void set_unk_entry (word_uid unk_word_id, const size_t num_unk_features, feature_array unk_features, const prob_weight lm_weight)
- bool is_num_entries_needed () const
- void set_num_entries (const size_t num_entries)
- tm_source_entry * begin_entry (const phrase_uid entry_id, const size_t num_elems)
- void finalize_entry (const phrase_uid entry_id)
- void finalize ()
- template<bool do_unk>
  tm_const_source_entry * get_source_entry (const phrase_uid entry_id) const
- bool is_unk_entry (tm_const_source_entry *entry) const
- void log_model_type_info () const

8.79.1 Detailed Description

This class represents a basic translation model implementation. The basic model is based on the fixed size hash map which is a self-implemented linear probing hash map also used in several tries. This basic model also does not store the phrases as is but rather the hash values thereof. So it is a hash based implementation which reduces memory but might occasionally result in collisions, the latter is not checked.

Definition at line 65 of file tm_basic_model.hpp.

8.79.2 Member Typedef Documentation

8.79.2.1 typedef fixed_size_hashmap<tm_source_entry, const phrase_uid &> uva::smt::bpbd::server::tm::models::tm_basic_model::tm_source_entry_map

Definition at line 68 of file tm_basic_model.hpp.

8.79.3 Constructor & Destructor Documentation

8.79.3.1 uva::smt::bpbd::server::tm::models::tm_basic_model::tm_basic_model ( ) [inline]

The basic class constructor

Definition at line 73 of file tm_basic_model.hpp.

8.79.3.2 uva::smt::bpbd::server::tm::models::tm_basic_model::~tm_basic_model ( ) [inline]

The basic destructor

Definition at line 79 of file tm_basic_model.hpp.

8.79.4 Member Function Documentation

8.79.4.1 tm_source_entry* uva::smt::bpbd::server::tm::models::tm_basic_model::begin_entry (const phrase_uid *entry_id,* const size_t *num_elems*) [inline]

Allows to open a new source entry, i.e. the entry for the new source phrase

Parameters

entry_id	the source phrase id for which the entry is to be started

Returns

the entry associated with the given id

Definition at line 158 of file tm_basic_model.hpp.

8.79.4.2 void uva::smt::bpbd::server::tm::models::tm_basic_model::finalize ( ) [inline]

This method is to be called when the translation model is fully read

Definition at line 200 of file tm_basic_model.hpp.

8.79.4.3 void uva::smt::bpbd::server::tm::models::tm_basic_model::finalize_entry (const phrase_uid *entry_id*) [inline]

Allows to finish an entry with the given id. The process of finishing might include many things but the purpose of it is to indicate that the source entry has been fully read. I.e. all the translations for the given source are processed.

Parameters

entry_id	the source phrase id for which the entry is to be finished.

Definition at line 188 of file tm_basic_model.hpp.

8.79.4.4 template<bool do_unk> tm_const_source_entry* uva::smt::bpbd::server::tm::models::tm_basic_model::get_source_entry (const phrase_uid *entry_id*) const [inline]

Allows to get the source entry for the given entry id

Parameters

do_unk	if true then if the entry is not present we return UNK if false then if the entry is not present we return NULL The default value is true.
entry_id	the source phrase id

Returns

the source phrase entry or UNK if the entry is not found

Definition at line 213 of file tm_basic_model.hpp.

8.79.4.5 bool uva::smt::bpbd::server::tm::models::tm_basic_model::is_num_entries_needed ( ) const [inline]

This method allows to detect if the number of entries (source phrases) is needed before the translation entries are being added.

Returns

true as this model type uses filed-size hash maps

Definition at line 136 of file tm_basic_model.hpp.

8.79.4.6 bool uva::smt::bpbd::server::tm::models::tm_basic_model::is_unk_entry (tm_const_source_entry * *entry*) const [inline]

Allows to check in the given entry is the UNK entry

Parameters

entry	the pointer to the entry to be tested

Returns

true if this is an UNK entry otherwise false

Definition at line 229 of file tm_basic_model.hpp.

8.79.4.7 void uva::smt::bpbd::server::tm::models::tm_basic_model::log_model_type_info ( ) const [inline]

Allows to log the model type info

Definition at line 236 of file tm_basic_model.hpp.

8.79.4.8 void uva::smt::bpbd::server::tm::models::tm_basic_model::set_num_entries (const size_t *num_entries*) [inline]

This method is needed to set the number of source phrase entries This is to be done before adding the translation entries to the model The memory of the map will be allocated by this class.

Parameters

num_entries	the number of source phrase entries

Definition at line 146 of file tm_basic_model.hpp.

8.79.4.9 void uva::smt::bpbd::server::tm::models::tm_basic_model::set_unk_entry (word_uid *unk_word_id,* const size_t *num_unk_features,* feature_array *unk_features,* const prob_weight *lm_weight*) [inline]

Should be called to add the unk entry to the model

Parameters

unk_word_id	the unknown word id from the Language Model
num_unk_features	the number of initialized unk features
unk_features	the unk entry features
lm_weight	the cost of the target (UNK) translation from the LM model

Definition at line 100 of file tm_basic_model.hpp.

The documentation for this class was generated from the following file:

- inc/server/tm/models/tm_basic_model.hpp

8.80 uva::smt::bpbd::server::tm::tm_configurator Class Reference

`#include <tm_configurator.hpp>`

Static Public Member Functions

- static void connect (const tm_parameters &params)
- static void disconnect ()
- static tm_query_proxy & allocate_query_proxy ()
- static void dispose_query_proxy (tm_query_proxy &query)

8.80.1 Detailed Description

This class represents a singleton that allows to configure the translation model and then issue a proxy object for performing the queries against it.

Definition at line 52 of file tm_configurator.hpp.

8.80.2 Member Function Documentation

8.80.2.1 static tm_query_proxy& uva::smt::bpbd::server::tm::tm_configurator::allocate_query_proxy ( ) [inline], [static]

Allows to return an instance of the query proxy, is to be returned by calling the dispose method.

Returns

an instance of the query executor.

Definition at line 91 of file tm_configurator.hpp.

8.80.2.2 static void uva::smt::bpbd::server::tm::tm_configurator::connect (const tm_parameters & *params*) [inline], [static]

This method allows to connect to the translation model. This method is to be called only once! The latter is not checked but is a must.

Parameters

params	the translation model parameters to be set, this class only stores the referent to the parameters.

Definition at line 62 of file tm_configurator.hpp.

8.80.2.3 static void uva::smt::bpbd::server::tm::tm_configurator::disconnect ( ) [inline],[static]

Allows to disconnect from the translation model.

Definition at line 76 of file tm_configurator.hpp.

8.80.2.4 static void uva::smt::bpbd::server::tm::tm_configurator::dispose_query_proxy (tm_query_proxy & *query*) [inline],[static]

Dispose the previously allocated query object

Parameters

query	the query to dispose

Definition at line 102 of file tm_configurator.hpp.

The documentation for this class was generated from the following files:

- inc/server/tm/tm_configurator.hpp
- src/server/tm/tm_configurator.cpp

8.81 uva::smt::bpbd::server::tm::tm_parameters Struct Reference

`#include <tm_parameters.hpp>`

Public Member Functions

- void finalize ()

Public Attributes

- string m_conn_string
- size_t m_num_lambdas
- float m_lambdas [NUM_TM_FEATURES]
- size_t m_num_unk_features
- float m_unk_features [NUM_TM_FEATURES]
- size_t m_trans_limit
- float m_min_tran_prob

8.81.1 Detailed Description

This structure stores the translation model parameters

Definition at line 53 of file tm_parameters.hpp.

8.81.2 Member Function Documentation

8.81.2.1 void uva::smt::bpbd::server::tm::tm_parameters::finalize ( ) [inline]

Allows to verify the parameters to be correct.

Definition at line 82 of file tm_parameters.hpp.

8.81.3 Member Data Documentation

8.81.3.1 string uva::smt::bpbd::server::tm::tm_parameters::m_conn_string

Definition at line 55 of file tm_parameters.hpp.

8.81.3.2 float uva::smt::bpbd::server::tm::tm_parameters::m_lambdas[NUM_TM_FEATURES]

Definition at line 61 of file tm_parameters.hpp.

8.81.3.3 float uva::smt::bpbd::server::tm::tm_parameters::m_min_tran_prob

Definition at line 77 of file tm_parameters.hpp.

8.81.3.4 size_t uva::smt::bpbd::server::tm::tm_parameters::m_num_lambdas

Definition at line 58 of file tm_parameters.hpp.

8.81.3.5 size_t uva::smt::bpbd::server::tm::tm_parameters::m_num_unk_features

Definition at line 64 of file tm_parameters.hpp.

8.81.3.6 size_t uva::smt::bpbd::server::tm::tm_parameters::m_trans_limit

Definition at line 71 of file tm_parameters.hpp.

8.81.3.7 float uva::smt::bpbd::server::tm::tm_parameters::m_unk_features[NUM_TM_FEATURES]

Definition at line 67 of file tm_parameters.hpp.

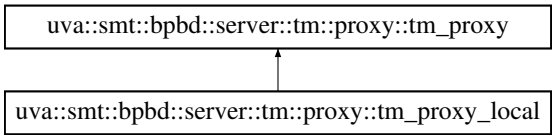
The documentation for this struct was generated from the following file:

- inc/server/tm/tm_parameters.hpp

8.82 uva::smt::bpbd::server::tm::proxy::tm_proxy Class Reference

`#include <tm_proxy.hpp>`

Inheritance diagram for uva::smt::bpbd::server::tm::proxy::tm_proxy:



Public Member Functions

- virtual void connect (const tm_parameters &params)=0
- virtual void disconnect ()=0
- virtual ~tm_proxy ()
- virtual tm_query_proxy & allocate_query_proxy ()=0
- virtual void dispose_query_proxy (tm_query_proxy &query)=0

8.82.1 Detailed Description

This is the translation model proxy interface class it allows to interact with any sort of local and remote models in a uniform way.

Definition at line 42 of file tm_proxy.hpp.

8.82.2 Constructor & Destructor Documentation

8.82.2.1 virtual uva::smt::bpbd::server::tm::proxy::tm_proxy::~tm_proxy ( ) [inline], [virtual]

The basic virtual destructor

Definition at line 59 of file tm_proxy.hpp.

8.82.3 Member Function Documentation

8.82.3.1 virtual tm_query_proxy& uva::smt::bpbd::server::tm::proxy::tm_proxy::allocate_query_proxy ( ) [pure virtual]

This method allows to get a query executor for the given trie

Returns

the trie query proxy object

Implemented in uva::smt::bpbd::server::tm::proxy::tm_proxy_local.

8.82.3.2 virtual void uva::smt::bpbd::server::tm::proxy::tm_proxy::connect (const tm_parameters & *params*) [pure virtual]

Allows to connect to the model object based on the given parameters

Parameters

params	the model parameters

Implemented in uva::smt::bpbd::server::tm::proxy::tm_proxy_local.

8.82.3.3 virtual void uva::smt::bpbd::server::tm::proxy::tm_proxy::disconnect ( ) [pure virtual]

Allows to disconnect from the trie

Implemented in uva::smt::bpbd::server::tm::proxy::tm_proxy_local.

8.82.3.4 virtual void uva::smt::bpbd::server::tm::proxy::tm_proxy::dispose_query_proxy (tm_query_proxy & *query*) [pure virtual]

Dispose the previously allocated query object

Parameters

query	the query to dispose

Implemented in uva::smt::bpbd::server::tm::proxy::tm_proxy_local.

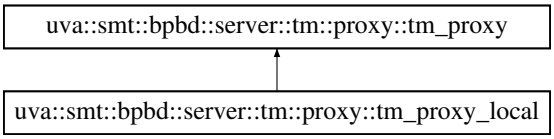
The documentation for this class was generated from the following file:

- inc/server/tm/proxy/tm_proxy.hpp

8.83 uva::smt::bpbd::server::tm::proxy::tm_proxy_local Class Reference

`#include <tm_proxy_local.hpp>`

Inheritance diagram for uva::smt::bpbd::server::tm::proxy::tm_proxy_local:



Public Member Functions

- tm_proxy_local ()
- virtual ~tm_proxy_local ()
- virtual void connect (const tm_parameters &params)
- virtual void disconnect ()
- virtual tm_query_proxy & allocate_query_proxy ()
- virtual void dispose_query_proxy (tm_query_proxy &query)

Protected Member Functions

- template<typename tm_builder_type , typename file_reader_type >
  void load_model_data (char const *model_name, const tm_parameters &params)

8.83.1 Detailed Description

This is the translation model proxy interface class it allows to interact with any sort of local and remote models in a uniform way.

Definition at line 60 of file tm_proxy_local.hpp.

8.83.2 Constructor & Destructor Documentation

8.83.2.1 uva::smt::bpbd::server::tm::proxy::tm_proxy_local::tm_proxy_local ( ) [inline]

The basic proxy constructor, currently does nothing except for default initialization

Definition at line 66 of file tm_proxy_local.hpp.

8.83.2.2 virtual uva::smt::bpbd::server::tm::proxy::tm_proxy_local::~tm_proxy_local ( ) [inline], [virtual]

The basic destructor

Definition at line 73 of file tm_proxy_local.hpp.

8.83.3 Member Function Documentation

8.83.3.1 virtual tm_query_proxy& uva::smt::bpbd::server::tm::proxy::tm_proxy_local::allocate_query_proxy ( ) [inline], [virtual]

See also

tm_proxy

Implements uva::smt::bpbd::server::tm::proxy::tm_proxy.

Definition at line 97 of file tm_proxy_local.hpp.

8.83.3.2 virtual void uva::smt::bpbd::server::tm::proxy::tm_proxy_local::connect (const tm_parameters & *params*) [inline], [virtual]

See also

tm_proxy

Implements uva::smt::bpbd::server::tm::proxy::tm_proxy.

Definition at line 81 of file tm_proxy_local.hpp.

8.83.3.3 virtual void uva::smt::bpbd::server::tm::proxy::tm_proxy_local::disconnect ( ) [inline], [virtual]

See also

tm_proxy

Implements uva::smt::bpbd::server::tm::proxy::tm_proxy.

Definition at line 90 of file tm_proxy_local.hpp.

8.83.3.4 virtual void uva::smt::bpbd::server::tm::proxy::tm_proxy_local::dispose_query_proxy (tm_query_proxy & *query*) [inline], [virtual]

Todo In the future we should just use a number of stack allocated objects in order to reduce the new/delete overhead

See also

tm_proxy

Implements uva::smt::bpbd::server::tm::proxy::tm_proxy.

Definition at line 106 of file tm_proxy_local.hpp.

8.83.3.5 template<typename tm_builder_type , typename file_reader_type > void uva::smt::bpbd::server::tm::proxy::tm_proxy_local::load_model_data (char const * *model_name,* const tm_parameters & *params*) [inline], [protected]

Allows to load the model into the instance of the selected container class

Todo Add the possibility to choose between the file readers from the command line!

Parameters

the	name of the model being loaded params the model parameters

Definition at line 119 of file tm_proxy_local.hpp.

The documentation for this class was generated from the following file:

- inc/server/tm/proxy/tm_proxy_local.hpp

8.84 uva::smt::bpbd::server::tm::models::tm_query< model_type > Class Template Reference

`#include <tm_query.hpp>`

Public Types

- typedef unordered_map< phrase_uid, tm_const_source_entry_ptr > query_map

Public Member Functions

- tm_query (const model_type &model)
- ~tm_query ()
- void execute (const phrase_uid uid, tm_const_source_entry_ptr &entry_ptr)
- tm_const_source_entry * get_source_entry (const phrase_uid uid)
- void get_st_uids (vector< phrase_uid > &st_uids) const

8.84.1 Detailed Description

template<typename model_type>class uva::smt::bpbd::server::tm::models::tm_query< model_type >

This class represents a query for the translation model In essence it is a map from the source phrases to the pointers to maps storing the translations in the target language. The query object is not re-usable at the moment as during the translation all source translations are retrieved once at the beginning of decoding. This query is based on hashing, i.e. internally the source phrase is stored as a hash value. This might occasionally cause collisions, but since this is a local issue it can be changed in the future.

Definition at line 64 of file tm_query.hpp.

8.84.2 Member Typedef Documentation

8.84.2.1 template<typename model_type> typedef unordered_map<phrase_uid, tm_const_source_entry_ptr> uva::smt::bpbd::server::tm::models::tm_query< model_type >::query_map

Definition at line 68 of file tm_query.hpp.

8.84.3 Constructor & Destructor Documentation

8.84.3.1 template<typename model_type> uva::smt::bpbd::server::tm::models::tm_query< model_type >::tm_query (const model_type & *model*) [inline]

The basic constructor

Definition at line 73 of file tm_query.hpp.

8.84.3.2 template<typename model_type> uva::smt::bpbd::server::tm::models::tm_query< model_type >::~tm_query ( ) [inline]

The basic destructor

Definition at line 79 of file tm_query.hpp.

8.84.4 Member Function Documentation

8.84.4.1 template<typename model_type> void uva::smt::bpbd::server::tm::models::tm_query< model_type >::execute (const phrase_uid *uid,* tm_const_source_entry_ptr & *entry_ptr*) [inline]

Allows to execute the translation query for the given source phrase. This query also keeps the local cache of retrieved source phrase translations.

Parameters

uid	[in] the source phrase uid
entry_ptr	[out] the reference to the source entry pointer which will be initialized with the found source entry.

Definition at line 90 of file tm_query.hpp.

8.84.4.2 template<typename model_type> tm_const_source_entry* uva::smt::bpbd::server::tm::models::tm_query< model_type >::get_source_entry (const phrase_uid *uid*) [inline]

Allows to get translations for the given source entry

Parameters

uid	the unique identifier of the source phrase

Returns

the pointer to the source entry or NULL if the translation is not found

Definition at line 122 of file tm_query.hpp.

8.84.4.3 template<typename model_type> void uva::smt::bpbd::server::tm::models::tm_query< model_type >::get_st_uids (vector< phrase_uid > & *st_uids*) const [inline]

Allows to get all the source/target phrase identifiers for the source target translation in this query. Must be called after the query is executed

Parameters

st_uids	the container for the source/target phrase identifiers

Definition at line 134 of file tm_query.hpp.

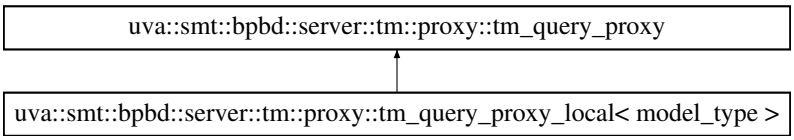
The documentation for this class was generated from the following file:

- inc/server/tm/models/tm_query.hpp

8.85 uva::smt::bpbd::server::tm::proxy::tm_query_proxy Class Reference

`#include <tm_query_proxy.hpp>`

Inheritance diagram for uva::smt::bpbd::server::tm::proxy::tm_query_proxy:



Public Member Functions

- virtual void execute (const phrase_uid uid, tm_const_source_entry_ptr &entry_ptr)=0
- virtual tm_const_source_entry * get_source_entry (const phrase_uid uid)=0
- virtual void get_st_uids (vector< phrase_uid > &st_uids) const =0
- virtual ~tm_query_proxy ()

8.85.1 Detailed Description

This class represents a translation query proxy interface class. It allows to interact with translation model queries in a uniform way.

Definition at line 46 of file tm_query_proxy.hpp.

8.85.2 Constructor & Destructor Documentation

8.85.2.1 virtual uva::smt::bpbd::server::tm::proxy::tm_query_proxy::~tm_query_proxy ( ) [inline], [virtual]

The basic virtual destructor

Definition at line 73 of file tm_query_proxy.hpp.

8.85.3 Member Function Documentation

8.85.3.1 virtual void uva::smt::bpbd::server::tm::proxy::tm_query_proxy::execute (const phrase_uid *uid*, tm_const_source_entry_ptr & *entry_ptr*) [pure virtual]

Allows to add the source phrase to the query.

Parameters

uid	the source phrase uid
entry_ptr	the reference to the source entry pointer to be set with the data

Implemented in uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local< model_type >.

8.85.3.2 virtual tm_const_source_entry* uva::smt::bpbd::server::tm::proxy::tm_query_proxy::get_source_entry (const phrase_uid *uid*) [pure virtual]

Allows retrieve the translations of the given source entry

Parameters

uid	the unique identifier of the source phrase

Returns

the pointer to the source entry or NULL if the translation is not found

Implemented in uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local< model_type >.

8.85.3.3 virtual void uva::smt::bpbd::server::tm::proxy::tm_query_proxy::get_st_uids (vector< phrase_uid > & *st_uids*) const [pure virtual]

Allows to get all the source/target phrase identifiers for the source target translation in this query. Must be called after the query is executed

Parameters

st_uids	the container for the source/target phrase identifiers

Implemented in uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local< model_type >.

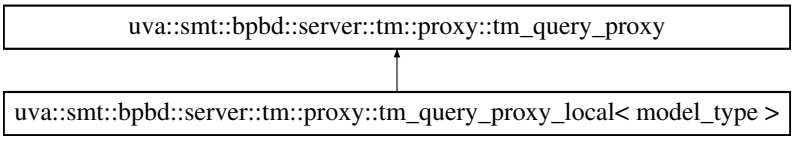
The documentation for this class was generated from the following file:

- inc/server/tm/proxy/tm_query_proxy.hpp

8.86 uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local< model_type > Class Template Reference

`#include <tm_query_proxy_local.hpp>`

Inheritance diagram for uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local< model_type >:



Public Member Functions

- tm_query_proxy_local (const model_type &model)
- virtual void execute (const phrase_uid uid, tm_const_source_entry_ptr &entry_ptr)
- virtual tm_const_source_entry * get_source_entry (const phrase_uid uid)
- virtual void get_st_uids (vector< phrase_uid > &st_uids) const
- virtual ~tm_query_proxy_local ()

8.86.1 Detailed Description

template<typename model_type>class uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local< model_type >

This is a local implementation of the translation model query This implementation works with the local translation model

Definition at line 47 of file tm_query_proxy_local.hpp.

8.86.2 Constructor & Destructor Documentation

8.86.2.1 template<typename model_type > uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local< model_type >::tm_query_proxy_local (const model_type & *model*) [inline]

The basic constructor that accepts the translation model reference to query to

Parameters

model	the translation model to query

Definition at line 54 of file tm_query_proxy_local.hpp.

8.86.2.2 template<typename model_type > virtual uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local< model_type >::~tm_query_proxy_local ( ) [inline], [virtual]

See also

tm_query_proxy

Definition at line 81 of file tm_query_proxy_local.hpp.

8.86.3 Member Function Documentation

8.86.3.1 template<typename model_type > virtual void uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local< model_type >::execute (const phrase_uid *uid,* tm_const_source_entry_ptr & *entry_ptr*) [inline], [virtual]

See also

tm_query_proxy

Implements uva::smt::bpbd::server::tm::proxy::tm_query_proxy.

Definition at line 60 of file tm_query_proxy_local.hpp.

8.86.3.2 template<typename model_type > virtual tm_const_source_entry* uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local< model_type >::get_source_entry (const phrase_uid *uid*) [inline], [virtual]

See also

tm_query_proxy

Implements uva::smt::bpbd::server::tm::proxy::tm_query_proxy.

Definition at line 67 of file tm_query_proxy_local.hpp.

8.86.3.3 template<typename model_type > virtual void uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local< model_type >::get_st_uids (vector< phrase_uid > & *st_uids*) const [inline], [virtual]

See also

tm_query_proxy

Implements uva::smt::bpbd::server::tm::proxy::tm_query_proxy.

Definition at line 74 of file tm_query_proxy_local.hpp.

The documentation for this class was generated from the following file:

- inc/server/tm/proxy/tm_query_proxy_local.hpp

8.87 uva::smt::bpbd::server::tm::models::tm_source_entry Class Reference

`#include <tm_source_entry.hpp>`

Public Member Functions

- tm_source_entry ()
- ~tm_source_entry ()
- bool has_translations () const
- phrase_uid get_source_uid ()
- void set_source_uid (phrase_uid s_uid)
- void begin (const size_t capacity)
- void finalize ()
- void add_target (const string &target, const phrase_uid target_uid, const size_t num_features, const prob_weight *features, const phrase_length num_words, const word_uid *word_ids, const prob_weight lm_weight)
- prob_weight get_min_cost () const
- bool has_target (const phrase_uid target_uid) const
- void get_st_uids (vector< phrase_uid > &st_uids) const
- bool operator== (const phrase_uid &phrase_uid) const
- bool operator== (const tm_source_entry &other) const
- size_t num_targets () const
- tm_const_target_entry * get_targets () const

8.87.1 Detailed Description

This is the source entry data structure that contains two things The source phrase uid, which is the unique identifier of the source string and the map storing the target translations. Note that the source phrase is not stored, this is to reduce memory consumption and improve speed. Similar as we did for the g2dm tried implementation for the language model.

Definition at line 65 of file tm_source_entry.hpp.

8.87.2 Constructor & Destructor Documentation

8.87.2.1 uva::smt::bpbd::server::tm::models::tm_source_entry::tm_source_entry ( ) [inline]

The basic constructor

Definition at line 71 of file tm_source_entry.hpp.

8.87.2.2 uva::smt::bpbd::server::tm::models::tm_source_entry::~tm_source_entry ( ) [inline]

The basic destructor

Definition at line 78 of file tm_source_entry.hpp.

8.87.3 Member Function Documentation

8.87.3.1 void uva::smt::bpbd::server::tm::models::tm_source_entry::add_target (const string & *target,* const **phrase_uid** *target_uid,* const size_t *num_features,* const **prob_weight** * *features,* const **phrase_length** *num_words,* const **word_uid** * *word_ids,* const **prob_weight** *lm_weight*) [inline]

Allows to add a new translation to the source entry for the given target phrase

Parameters

target	the target phrase string
target_uid	the uid of the target phrase
num_features	the number of features in the next array
weights	the features to put into the entry
num_words	the number of words in the target translation
word_ids	the LM word ids for the target phrase
lm_weight	the cost of the target translation from the LM model

Definition at line 147 of file tm_source_entry.hpp.

8.87.3.2 void uva::smt::bpbd::server::tm::models::tm_source_entry::begin (const size_t *capacity*) [inline]

Should be called to start the source entry, i.e. initialize the memory

Parameters

capacity	the number of translations for this entry

Definition at line 119 of file tm_source_entry.hpp.

8.87.3.3 void uva::smt::bpbd::server::tm::models::tm_source_entry::finalize ( ) [inline]

Should be called to indicate that this source entry is finished, i.e. all the translations have been set.

Definition at line 129 of file tm_source_entry.hpp.

8.87.3.4 prob_weight uva::smt::bpbd::server::tm::models::tm_source_entry::get_min_cost ( ) const [inline]

Allows to get the minimum translation cost for the given source phrase, i.e what we have is: "log_10(maximum_↩ t(P_tm(t|s)*P_lm(t)))

Returns

the minimum translation cost for the given source phrase

Definition at line 172 of file tm_source_entry.hpp.

8.87.3.5 phrase_uid uva::smt::bpbd::server::tm::models::tm_source_entry::get_source_uid ( ) [inline]

Allows to get the source phrase id

Returns

the source phrase id

Definition at line 103 of file tm_source_entry.hpp.

8.87.3.6 void uva::smt::bpbd::server::tm::models::tm_source_entry::get_st_uids (vector< phrase_uid > & *st_uids*) const [inline]

Allows to get all the source/target phrase identifiers for the source target translation in this query.

Parameters

st_uids	the container for the source/target phrase identifiers

Definition at line 216 of file tm_source_entry.hpp.

8.87.3.7 tm_const_target_entry* uva::smt::bpbd::server::tm::models::tm_source_entry::get_targets ( ) const [inline]

Allows to get an array of of target entries, if any

Returns

the pointer to the first target entry, or NULL if none

Definition at line 252 of file tm_source_entry.hpp.

8.87.3.8 bool uva::smt::bpbd::server::tm::models::tm_source_entry::has_target (const phrase_uid *target_uid*) const [inline]

Allows to check if the translation of the given target is present. NOTE: This check is not optimal a better data structure for storing entries might be needed, although this method is only used when building the Reordering model.

Parameters

target_uid	the unique identifier of the taret

Returns

true if the target is known, otherwise false

Definition at line 184 of file tm_source_entry.hpp.

8.87.3.9 bool uva::smt::bpbd::server::tm::models::tm_source_entry::has_translations ( ) const [inline]

Allows to detect whether the given source entry is for the UNK phrase or not. The UNK phrase is the phrase with an unknown translation

Returns

false if the source entry is for the UNK phrase, otherwise false

Definition at line 90 of file tm_source_entry.hpp.

8.87.3.10 size_t uva::smt::bpbd::server::tm::models::tm_source_entry::num_targets ( ) const [inline]

Allows to get the number of target entries

Returns

the number of target entries, or 0 if none

Definition at line 244 of file tm_source_entry.hpp.

8.87.3.11 bool uva::smt::bpbd::server::tm::models::tm_source_entry::operator== (const phrase_uid & *phrase_uid*) const [inline]

The comparison operator, allows to compare source entries

Parameters

phrase_uid	the unique identifier of the source entry to compare with

Returns

true if the provided uid is equal to the uid of this entry, otherwise false

Definition at line 227 of file tm_source_entry.hpp.

8.87.3.12 bool uva::smt::bpbd::server::tm::models::tm_source_entry::operator== (const tm_source_entry & *other*) const [inline]

The comparison operator, allows to compare source entries

Parameters

other	the the source entry to compare with

Returns

true if the uid of the provided entry is equal to the uid of this entry, otherwise false

Definition at line 236 of file tm_source_entry.hpp.

8.87.3.13 void uva::smt::bpbd::server::tm::models::tm_source_entry::set_source_uid (phrase_uid *s_uid*) [inline]

Allows to set the source phrase id

Parameters

s_uid	the source phrase id

Definition at line 111 of file tm_source_entry.hpp.

The documentation for this class was generated from the following file:

- inc/server/tm/models/tm_source_entry.hpp

8.88 uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features > Class Template Reference

`#include <tm_target_entry.hpp>`

Public Member Functions

- tm_target_entry_temp ()
- ~tm_target_entry_temp ()
- void set_data (const phrase_uid source_uid, const string &target_phrase, const phrase_uid target_uid, const size_t num_features, const float *features, const phrase_length num_words, const word_uid *word_ids)
- bool is_unk_trans () const
- const string & get_target_phrase () const
- const phrase_uid get_st_uid () const
- const prob_weight get_total_weight () const
- const prob_weight get_t_c_s () const
- phrase_length get_num_words () const
- const word_uid * get_word_ids () const

Static Public Attributes

- static constexpr uint8_t NUM_FEATURES = max_num_features
- static const phrase_uid UNKNOWN_TARGET_ENTRY_UID = combine_phrase_uids(UNKNOWN_PHRASE_ID, UNKNOWN_PHRASE_ID)

Protected Member Functions

- void set_features (const size_t num_features, const prob_weight *features)

8.88.1 Detailed Description

template<uint8_t max_num_features>class uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features >

This structure represents the translation data, i.e. the the target phrase plus the probability weights. See: http://www.statmt.org/moses/?n=FactoredTraining.ScorePhrases for more details on the weights. Note that for this entry we have a uid that is a unique identifier of the target phrase string. The latter can be a hash value but then there is a possibility for the hash collisions

Definition at line 66 of file tm_target_entry.hpp.

8.88.2 Constructor & Destructor Documentation

8.88.2.1 template<uint8_t max_num_features> uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features >::tm_target_entry_temp ( ) [inline]

The basic constructor

Definition at line 77 of file tm_target_entry.hpp.

8.88.2.2 template<uint8_t max_num_features> uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features >::~tm_target_entry_temp ( ) [inline]

The basic destructor

Definition at line 85 of file tm_target_entry.hpp.

8.88.3 Member Function Documentation

8.88.3.1 template<uint8_t max_num_features> phrase_length uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features >::get_num_words ( ) const [inline]

Allows to get the number of words in the target translation

Returns

the number of words

Definition at line 172 of file tm_target_entry.hpp.

8.88.3.2 template<uint8_t max_num_features> const phrase_uid uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features >::get_st_uid ( ) const [inline]

Allows to retrieve the source/target phrase pair uid

Returns

the source/target phrase pair uid

Definition at line 146 of file tm_target_entry.hpp.

8.88.3.3 template<uint8_t max_num_features> const prob_weight uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features >::get_t_c_s ( ) const [inline]

Allows to get the value of the third feature which is the log10(p(e|f))

Returns

the value of the third feature which is the log10(p(e|f))

Definition at line 163 of file tm_target_entry.hpp.

8.88.3.4 template<uint8_t max_num_features> const string& uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features >::get_target_phrase ( ) const [inline]

Allows to get the target phrase

Returns

the reference to the const target phrase

Definition at line 138 of file tm_target_entry.hpp.

8.88.3.5 template<uint8_t max_num_features> const prob_weight uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features >::get_total_weight ( ) const [inline]

Allows to get the total weight of the entry, the sum of features that are turned into log10 scale.

Returns

the total weight of the entry, the sum of feature weights

Definition at line 155 of file tm_target_entry.hpp.

8.88.3.6 template<uint8_t max_num_features> const word_uid* uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features >::get_word_ids ( ) const [inline]

This method allows to get the

Returns

an array of word ids of the target phrase, the length must be equal to LM_QUERY_LENGTH_MAX

Definition at line 180 of file tm_target_entry.hpp.

8.88.3.7 template<uint8_t max_num_features> bool uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features >::is_unk_trans ( ) const [inline]

Allows to check whether this is an unknown translation

Returns

true if this is UNK translation, otherwise false

Definition at line 130 of file tm_target_entry.hpp.

8.88.3.8 template<uint8_t max_num_features> void uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features >::set_data (const phrase_uid *source_uid*, const string & *target_phrase*, const phrase_uid *target_uid*, const size_t *num_features*, const float * *features*, const phrase_length *num_words*, const word_uid * *word_ids*) [inline]

Allows to set the target phrase and its id

Parameters

source_uid	store the source uid for being combined with the target phrase into the source/target pair uid
target_phrase	the target phrase
target_uid	the uid of the target phrase
num_features	the number of features to be set, already in the log10 scale
features	the weights to be set into the entry
num_words	the number of words in the target translation
word_ids	the LM word ids for the target phrase

Definition at line 104 of file tm_target_entry.hpp.

8.88.3.9 template<uint8_t max_num_features> void uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features >::set_features (const size_t *num_features*, const prob_weight * *features*) [inline], [protected]

Allows to set the weights into the target entry.

Todo Get rid of magic constants here!

Parameters

num_features	the number of features to be set, already in the log10 scale
features	the weights to be set into the entry This is an array of translation weights, as we have here : features[0] = p(f|e); features[1] = lex(p(f|e)); features[2] = p(e|f); features[3] = lex(p(e|f)); features[4] = phrase penalty; // optional

Definition at line 198 of file tm_target_entry.hpp.

8.88.4 Member Data Documentation

8.88.4.1 template<uint8_t max_num_features> constexpr uint8_t uva::smt::bpbd::server::tm::models::tm_target_entry_temp< num_features >::NUM_FEATURES = max_num_features [static]

Definition at line 69 of file tm_target_entry.hpp.

8.88.4.2 template<uint8_t max_num_features> const phrase_uid uva::smt::bpbd::server::tm::models::tm_target_entry_temp< num_features >::UNKNOWN_TARGET_ENTRY_UID = combine_phrase_uids(UNKNOWN_PHRASE_ID, UNKNOWN_PHRASE_ID) [static]

Definition at line 72 of file tm_target_entry.hpp.

The documentation for this class was generated from the following files:

- inc/server/tm/models/tm_target_entry.hpp
- src/server/tm/models/tm_target_entry.cpp

8.89 uva::smt::bpbd::client::trans_job Struct Reference

`#include <trans_job.hpp>`

Public Member Functions

- trans_job ()
- virtual ~trans_job ()

Public Attributes

- uint32_t m_num_sentences
- trans_job_request_ptr m_request
- trans_job_response_ptr m_response
- trans_job_status m_status

8.89.1 Detailed Description

This structure is used for storing the translation job data

Definition at line 51 of file trans_job.hpp.

8.89.2 Constructor & Destructor Documentation

8.89.2.1 uva::smt::bpbd::client::trans_job::trans_job ( ) [inline]

The basic constructor that does default-initialization of the structure fields

Definition at line 56 of file trans_job.hpp.

8.89.2.2 virtual uva::smt::bpbd::client::trans_job::~trans_job ( ) [inline], [virtual]

The basic destructor that allows to delete the dynamically allocated data pointed by the structure fields

Definition at line 65 of file trans_job.hpp.

8.89.3 Member Data Documentation

8.89.3.1 uint32_t uva::smt::bpbd::client::trans_job::m_num_sentences

Definition at line 75 of file trans_job.hpp.

8.89.3.2 trans_job_request_ptr uva::smt::bpbd::client::trans_job::m_request

Definition at line 77 of file trans_job.hpp.

8.89.3.3 trans_job_response_ptr uva::smt::bpbd::client::trans_job::m_response

Definition at line 79 of file trans_job.hpp.

8.89.3.4 trans_job_status uva::smt::bpbd::client::trans_job::m_status

Definition at line 81 of file trans_job.hpp.

The documentation for this struct was generated from the following file:

- inc/client/trans_job.hpp

8.90 uva::smt::bpbd::server::trans_job Class Reference

`#include <trans_job.hpp>`

Public Types

- typedef function< void(trans_job_ptr trans_job) > done_job_notifier
- typedef vector< trans_task_ptr > tasks_list_type
- typedef tasks_list_type::iterator tasks_iter_type
- typedef tasks_list_type::const_iterator tasks_const_iter_type

Public Member Functions

- trans_job (trans_job_request_ptr request_ptr)
- void set_done_job_notifier (done_job_notifier notify_job_done_func)
- virtual ~trans_job ()

- const session_id_type get_session_id () const
- const job_id_type get_job_id () const
- const tasks_list_type & get_tasks ()
- virtual const trans_job_code get_code () const
- virtual const string & get_text () const
- void cancel ()

Protected Member Functions

- bool is_job_finished ()
- void notify_task_done (const trans_task_ptr &task)
- void combine_job_result ()

8.90.1 Detailed Description

This class represents the translation job. Each translation job belongs to a session and contains a translation request. Every translation request is a text consisting of multiple sentences. The translation job therefore splits this request into a number of translation tasks each of which translates one sentence.

Definition at line 61 of file trans_job.hpp.

8.90.2 Member Typedef Documentation

8.90.2.1 typedef function<void(trans_job_ptr trans_job)> uva::smt::bpbd::server::trans_job::done_job_notifier

Definition at line 65 of file trans_job.hpp.

8.90.2.2 typedef tasks_list_type::const_iterator uva::smt::bpbd::server::trans_job::tasks_const_iter_type

Definition at line 70 of file trans_job.hpp.

8.90.2.3 typedef tasks_list_type::iterator uva::smt::bpbd::server::trans_job::tasks_iter_type

Definition at line 69 of file trans_job.hpp.

8.90.2.4 typedef vector<trans_task_ptr> uva::smt::bpbd::server::trans_job::tasks_list_type

Definition at line 68 of file trans_job.hpp.

8.90.3 Constructor & Destructor Documentation

8.90.3.1 uva::smt::bpbd::server::trans_job::trans_job (trans_job_request_ptr *request_ptr*) [inline]

The basic constructor allowing to initialize the main class constants

Parameters

session_id	the id of the session from which the translation request is received

job_id	the translation job id
task_ids	the list of task ids from which this job consists of

Definition at line 78 of file trans_job.hpp.

8.90.3.2 virtual uva::smt::bpbd::server::trans_job::~trans_job ( ) [inline], [virtual]

The basic destructor

Definition at line 114 of file trans_job.hpp.

8.90.4 Member Function Documentation

8.90.4.1 void uva::smt::bpbd::server::trans_job::cancel ( ) [inline]

Allows to cancel the given translation job by telling all the translation tasks to stop.

Definition at line 177 of file trans_job.hpp.

8.90.4.2 void uva::smt::bpbd::server::trans_job::combine_job_result ( ) [inline], [protected]

Allows to compile the end job result, e.g. based on the task results, come up with the job's result code and the translated text.

Definition at line 251 of file trans_job.hpp.

8.90.4.3 virtual const trans_job_code uva::smt::bpbd::server::trans_job::get_code ( ) const [inline], [virtual]

Allows to retrieve the translation task result code

Returns

the translation task result code

Definition at line 162 of file trans_job.hpp.

8.90.4.4 const job_id_type uva::smt::bpbd::server::trans_job::get_job_id ( ) const [inline]

Allows to retrieve the job id

Returns

the job id

Definition at line 146 of file trans_job.hpp.

8.90.4.5 const session_id_type uva::smt::bpbd::server::trans_job::get_session_id ( ) const [inline]

Allows to retrieve the session id

Returns

the session id

Definition at line 138 of file trans_job.hpp.

8.90.4.6 const tasks_list_type& uva::smt::bpbd::server::trans_job::get_tasks ( ) [inline]

Allows to get the list of translation tasks

Returns

the list of translation tasks of this job

Definition at line 154 of file trans_job.hpp.

8.90.4.7 virtual const string& uva::smt::bpbd::server::trans_job::get_text ( ) const [inline], [virtual]

Allows to retrieve the translation task result text

Returns

the translation task result text

Definition at line 170 of file trans_job.hpp.

8.90.4.8 bool uva::smt::bpbd::server::trans_job::is_job_finished ( ) [inline], [protected]

Allows to check if the job is finished by checking the number of finished tasks. The check is synchronized.

Returns

true if all the job's tasks are finished, otherwise false

Definition at line 195 of file trans_job.hpp.

8.90.4.9 void uva::smt::bpbd::server::trans_job::notify_task_done (const trans_task_ptr & *task*) [inline], [protected]

Is used from the translation task to notify the translation job that the task is ready. This method is thread safe.

Todo {Do a strict check on the tasks reporting to be finished, these should be the ones from the m_tasks list and they must report themselves only ones. (Optional - for safety).}

Parameters

task	the translation task that is finished

Definition at line 216 of file trans_job.hpp.

8.90.4.10 void uva::smt::bpbd::server::trans_job::set_done_job_notifier (done_job_notifier *notify_job_done_func*) [inline]

Allows to set the function that should be called when the job is done

Parameters

notify_job_done_func	

Definition at line 107 of file trans_job.hpp.

The documentation for this class was generated from the following file:

- inc/server/trans_job.hpp

8.91 uva::smt::bpbd::common::messaging::trans_job_code Class Reference

`#include <trans_job_code.hpp>`

Public Types

- enum values {
  RESULT_UNDEFINED = 0, RESULT_OK = RESULT_UNDEFINED + 1, RESULT_ERROR = RESULT_OK + 1, RESULT_CANCELED = RESULT_ERROR + 1,
  RESULT_PARTIAL = RESULT_CANCELED + 1, size = RESULT_PARTIAL + 1 }

Public Member Functions

- trans_job_code (const values code)
- trans_job_code (const int32_t code_val)
- trans_job_code ()
- void operator= (const values &code)
- bool operator== (const values &code) const
- bool operator< (const values &code) const
- operator string () const
- operator int () const
- const char *const str () const
- values val ()

8.91.1 Detailed Description

This class represents the translation job result code. This class is used on client and server side. It represents the server-side status of a translation job and a translation taks.

Definition at line 49 of file trans_job_code.hpp.

8.91.2 Member Enumeration Documentation

8.91.2.1 enum uva::smt::bpbd::common::messaging::trans_job_code::values

Stores the translation job result codes, currently there is just two results possible, the job is done - OK; or there was some error - ERROR

Enumerator

RESULT_UNDEFINED

RESULT_OK

RESULT_ERROR

RESULT_CANCELED

RESULT_PARTIAL

size

Definition at line 57 of file trans_job_code.hpp.

8.91.3 Constructor & Destructor Documentation

8.91.3.1 uva::smt::bpbd::common::messaging::trans_job_code::trans_job_code (const values *code*) [inline]

The basic constructor that allows to initialize the value with the code

Parameters

code	the code value to initialize with

Definition at line 70 of file trans_job_code.hpp.

8.91.3.2 uva::smt::bpbd::common::messaging::trans_job_code::trans_job_code (const int32_t *code_val*) [inline]

The basic constructor that allows to initialize the value from an integer

Parameters

code_val	the code value to initialize with

Definition at line 77 of file trans_job_code.hpp.

8.91.3.3 uva::smt::bpbd::common::messaging::trans_job_code::trans_job_code ( ) [inline]

The basic constructor that creates an undefined value

Definition at line 89 of file trans_job_code.hpp.

8.91.4 Member Function Documentation

8.91.4.1 uva::smt::bpbd::common::messaging::trans_job_code::operator int ( ) const [inline]

The operator allowing to convert the value to an integer

Returns

the the integer value

Definition at line 128 of file trans_job_code.hpp.

8.91.4.2 uva::smt::bpbd::common::messaging::trans_job_code::operator string ( ) const [inline]

The operator allowing to convert the value to string

Returns

the string representation of the code

Definition at line 120 of file trans_job_code.hpp.

8.91.4.3 bool uva::smt::bpbd::common::messaging::trans_job_code::operator< (const values & *code*) const [inline]

Overloading the comparison operator for the code

Parameters

code	the code to compare with

Definition at line 112 of file trans_job_code.hpp.

8.91.4.4 void uva::smt::bpbd::common::messaging::trans_job_code::operator= (const values & *code*) [inline]

Overloading the assignment operator for the code

Parameters

code	the code to set

Definition at line 96 of file trans_job_code.hpp.

8.91.4.5 bool uva::smt::bpbd::common::messaging::trans_job_code::operator== (const values & *code*) const [inline]

Overloading the equality operator for the code

Parameters

code	the code to check equality with

Definition at line 104 of file trans_job_code.hpp.

8.91.4.6 const char *const uva::smt::bpbd::common::messaging::trans_job_code::str ( ) const

Allows to get the job code string for reporting

Returns

the job code string

Definition at line 51 of file trans_job_code.cpp.

8.91.4.7 values uva::smt::bpbd::common::messaging::trans_job_code::val ( ) [inline]

Returns the stored code value

Returns

the stored code value

Definition at line 142 of file trans_job_code.hpp.

The documentation for this class was generated from the following files:

- inc/common/messaging/trans_job_code.hpp
- src/common/messaging/trans_job_code.cpp

8.92 uva::smt::bpbd::server::trans_job_pool Class Reference

`#include <trans_job_pool.hpp>`

Public Types

- typedef function< void(trans_job_ptr trans_job) > finished_job_notifier
- typedef std::map< job_id_type, trans_job_ptr > jobs_map_type
- typedef jobs_map_type::iterator jobs_map_iter_type
- typedef std::map< session_id_type, jobs_map_type > sessions_map_type
- typedef sessions_map_type::iterator sessions_map_iter_type
- typedef vector< trans_job_ptr > jobs_list_type
- typedef jobs_list_type::iterator jobs_list_iter_type

Public Member Functions

- trans_job_pool (const size_t num_threads)
- virtual ~trans_job_pool ()
- void stop ()
- void set_num_threads (const size_t num_threads)
- void report_run_time_info ()
- void set_job_result_setter (finished_job_notifier notify_job_finished_func)
- void plan_new_job (trans_job_ptr trans_job)
- void cancel_jobs (const session_id_type session_id)

Protected Member Functions

- void cancel_all_jobs ()
- void add_job (trans_job_ptr trans_job)
- void delete_job (trans_job_ptr trans_job)
- bool is_stop_running ()
- void wake_up_jobs_thread ()
- void notify_job_done (trans_job_ptr trans_job)
- void process_finished_jobs ()

8.92.1 Detailed Description

This class is used to schedule the translation jobs. Each translation job consists of a number of sentences to translate. Each sentence will be translated in its own thread with its own decoder instance. The job of this class is to split the translation job into a number of translation tasks and schedule them. This class is synchronized and has its own thread to schedule the translation tasks.

Definition at line 66 of file trans_job_pool.hpp.

8.92.2 Member Typedef Documentation

8.92.2.1 typedef function<void(trans_job_ptr trans_job) > uva::smt::bpbd::server::trans_job_pool::finished_job_notifier

Definition at line 70 of file trans_job_pool.hpp.

8.92.2.2 typedef jobs_list_type::iterator uva::smt::bpbd::server::trans_job_pool::jobs_list_iter_type

Definition at line 80 of file trans_job_pool.hpp.

8.92.2.3 typedef vector<trans_job_ptr> uva::smt::bpbd::server::trans_job_pool::jobs_list_type

Definition at line 79 of file trans_job_pool.hpp.

8.92.2.4 typedef jobs_map_type::iterator uva::smt::bpbd::server::trans_job_pool::jobs_map_iter_type

Definition at line 74 of file trans_job_pool.hpp.

8.92.2.5 typedef std::map<job_id_type, trans_job_ptr> uva::smt::bpbd::server::trans_job_pool::jobs_map_type

Definition at line 73 of file trans_job_pool.hpp.

8.92.2.6 typedef sessions_map_type::iterator uva::smt::bpbd::server::trans_job_pool::sessions_map_iter_type

Definition at line 76 of file trans_job_pool.hpp.

8.92.2.7 typedef std::map<session_id_type, jobs_map_type> uva::smt::bpbd::server::trans_job_pool::sessions_map_type

Definition at line 75 of file trans_job_pool.hpp.

8.92.3 Constructor & Destructor Documentation

8.92.3.1 uva::smt::bpbd::server::trans_job_pool::trans_job_pool (const size_t *num_threads*) [inline]

The basic constructor, starts the finished jobs processing thread.

Parameters

num_threads	the number of translation threads to run

Definition at line 86 of file trans_job_pool.hpp.

8.92.3.2 virtual uva::smt::bpbd::server::trans_job_pool::~trans_job_pool ( ) [inline], [virtual]

he basic destructor

Definition at line 94 of file trans_job_pool.hpp.

8.92.4 Member Function Documentation

8.92.4.1 void uva::smt::bpbd::server::trans_job_pool::add_job (trans_job_ptr *trans_job*) [inline], [protected]

Allows to add a new job to the administration. In case the session is not known or the job id is already in use an exception is thrown. Also the job count is incremented

Todo {Later, the tasks pool shall be chosen based on the source and target language. This is for when a server can translate from multiple languages to multiple languages.}

Parameters

trans_job	the job to be added to the administration

Definition at line 253 of file trans_job_pool.hpp.

8.92.4.2 void uva::smt::bpbd::server::trans_job_pool::cancel_all_jobs ( ) [inline], [protected]

Allows to cancel all the currently running translation jobs in the server

Definition at line 230 of file trans_job_pool.hpp.

8.92.4.3 void uva::smt::bpbd::server::trans_job_pool::cancel_jobs (const session_id_type *session_id*) [inline]

Allows to cancel all translation jobs for the given session id.

Parameters

session_id	the session id to cancel the jobs for

Definition at line 201 of file trans_job_pool.hpp.

8.92.4.4 void uva::smt::bpbd::server::trans_job_pool::delete_job (trans_job_ptr *trans_job*) [inline], [protected]

Allows to delete the given job from the administration, decrement the jobs count and destroy the job object.

Parameters

trans_job	the job to be deleted

Definition at line 288 of file trans_job_pool.hpp.

8.92.4.5 bool uva::smt::bpbd::server::trans_job_pool::is_stop_running ( ) [inline], [protected]

Allows to check if the finished jobs processing loop has to stop.

Returns

true if the finished jobs processing loop has to stop, otherwise false

Definition at line 324 of file trans_job_pool.hpp.

8.92.4.6 void uva::smt::bpbd::server::trans_job_pool::notify_job_done (trans_job_ptr *trans_job*) [inline], [protected]

Allows notify the job pool that the given job is done.

Parameters

trans_job	the pointer to the finished translation job

Definition at line 363 of file trans_job_pool.hpp.

8.92.4.7 void uva::smt::bpbd::server::trans_job_pool::plan_new_job (trans_job_ptr *trans_job*) [inline]

Allows to schedule a new translation job. The execution of the job is deferred and asynchronous. trans_job the translation job to be scheduled

Definition at line 180 of file trans_job_pool.hpp.

8.92.4.8 void uva::smt::bpbd::server::trans_job_pool::process_finished_jobs ( ) [inline], [protected]

Allows to process the finished translation jobs

Definition at line 383 of file trans_job_pool.hpp.

8.92.4.9 void uva::smt::bpbd::server::trans_job_pool::report_run_time_info ( ) [inline]

Allows to report the runtime information.

Definition at line 154 of file trans_job_pool.hpp.

8.92.4.10 void uva::smt::bpbd::server::trans_job_pool::set_job_result_setter (finished_job_notifier *notify_job_finished_func*) [inline]

Allows to set the response sender function for sending the replies to the client

Parameters

notify_job_↩ finished_func	the setter functional to be set

Definition at line 171 of file trans_job_pool.hpp.

8.92.4.11 void uva::smt::bpbd::server::trans_job_pool::set_num_threads (const size_t *num_threads*) [inline]

Allows to set the new number of worker threads. This operation should be safe as the new threads are just added to the list and the deleted ones are let to finish their translation task execution.

Parameters

num_threads	the new number of worker threads

Definition at line 147 of file trans_job_pool.hpp.

8.92.4.12 void uva::smt::bpbd::server::trans_job_pool::stop ( ) [inline]

Allows to stop all the running jobs and try to send all the responses and then exit

Definition at line 101 of file trans_job_pool.hpp.

8.92.4.13 void uva::smt::bpbd::server::trans_job_pool::wake_up_jobs_thread ( ) [inline], [protected]

Allows to wake up the jobs thread.

Definition at line 352 of file trans_job_pool.hpp.

The documentation for this class was generated from the following file:

- inc/server/trans_job_pool.hpp

8.93 uva::smt::bpbd::common::messaging::trans_job_request Class Reference

`#include <trans_job_request.hpp>`

Public Member Functions

- trans_job_request (const string &message)
- trans_job_request (const job_id_type job_id, const string &source_lang, const string &text, const string &target_lang)
- void de_serialize (const string &message)
- const string serialize () const
- void set_session_id (const session_id_type session_id)
- const session_id_type get_session_id () const
- const job_id_type get_job_id () const
- const string get_source_lang () const
- const string get_target_lang () const
- const string & get_text () const

Static Public Attributes

- static constexpr char HEADER_DELIMITER = ':'
- static constexpr char NEW_LINE_HEADER_ENDING = '\n'
- static constexpr char TEXT_SENTENCE_DELIMITER = '\n'

8.93.1 Detailed Description

This class represents the translation request message.

Definition at line 53 of file trans_job_request.hpp.

8.93.2 Constructor & Destructor Documentation

8.93.2.1 uva::smt::bpbd::common::messaging::trans_job_request::trans_job_request (const string & *message*) [inline]

This is the basic class constructor that accepts the original client message to parse. This constructor is to be used on the server to de-serialize the translation request.

Parameters

message	the client translation request to be parsed

Definition at line 67 of file trans_job_request.hpp.

8.93.2.2 uva::smt::bpbd::common::messaging::trans_job_request::trans_job_request (const job_id_type *job_id,* const string & *source_lang,* const string & *text,* const string & *target_lang*) [inline]

This is the basic class constructor that accepts the translation job id, the translation text and source and target language strings.

Parameters

job_id	the translation job id
source_lang	the source language string
text	the text in the source language to translate
target_lang	the target language string

Definition at line 81 of file trans_job_request.hpp.

8.93.3 Member Function Documentation

8.93.3.1 void uva::smt::bpbd::common::messaging::trans_job_request::de_serialize (const string & *message*) [inline]

Allows to de-serialize the job request from a string

Parameters

message	the string representation of the translation job request

Definition at line 91 of file trans_job_request.hpp.

8.93.3.2 const job_id_type uva::smt::bpbd::common::messaging::trans_job_request::get_job_id ( ) const [inline]

Allows to get the client-issued job id

Returns

the client-issued job id

Definition at line 164 of file trans_job_request.hpp.

8.93.3.3 const session_id_type uva::smt::bpbd::common::messaging::trans_job_request::get_session_id ( ) const [inline]

Allows to get the translation session id. This method to be used on the client, for the sake of storing the session id by the translation job request class.

Returns

the session id issued by the server

Definition at line 156 of file trans_job_request.hpp.

8.93.3.4 const string uva::smt::bpbd::common::messaging::trans_job_request::get_source_lang ( ) const [inline]

Allows to get the translation job source language

Returns

the translation job source language

Definition at line 172 of file trans_job_request.hpp.

8.93.3.5 const string uva::smt::bpbd::common::messaging::trans_job_request::get_target_lang ( ) const [inline]

Allows to get the translation job target language

Returns

the translation job target language

Definition at line 180 of file trans_job_request.hpp.

8.93.3.6 const string& uva::smt::bpbd::common::messaging::trans_job_request::get_text ( ) const [inline]

Allows to get the translation job text. This is either the text translated into the target language or the error message for the case of failed translation job request.

Returns

the translation job text

Definition at line 190 of file trans_job_request.hpp.

8.93.3.7 const string uva::smt::bpbd::common::messaging::trans_job_request::serialize ( ) const [inline]

Allows to serialize the job request into a string

Returns

the string representation of the translation job request

Definition at line 130 of file trans_job_request.hpp.

8.93.3.8 void uva::smt::bpbd::common::messaging::trans_job_request::set_session_id (const session_id_type *session_id*) [inline]

Allows to set the translation session id. This method to be used on the client, for the sake of storing the session id by the translation job request class.

Parameters

session_id	the session id issued by the server

Definition at line 146 of file trans_job_request.hpp.

8.93.4 Member Data Documentation

8.93.4.1 constexpr char uva::smt::bpbd::common::messaging::trans_job_request::HEADER_DELIMITER = ':' [static]

Definition at line 56 of file trans_job_request.hpp.

8.93.4.2 constexpr char uva::smt::bpbd::common::messaging::trans_job_request::NEW_LINE_HEADER_ENDING = '\n' [static]

Definition at line 57 of file trans_job_request.hpp.

8.93.4.3 constexpr char uva::smt::bpbd::common::messaging::trans_job_request::TEXT_SENTENCE_DELIMITER = '\n' [static]

Definition at line 58 of file trans_job_request.hpp.

The documentation for this class was generated from the following file:

- inc/common/messaging/trans_job_request.hpp

8.94 uva::smt::bpbd::common::messaging::trans_job_response Class Reference

`#include <trans_job_response.hpp>`

Public Member Functions

- trans_job_response ()
- trans_job_response (const string &message)
- trans_job_response (const job_id_type job_id, const trans_job_code code, const string &text)
- void de_serialize (const string &message)
- const string serialize ()
- const job_id_type get_job_id () const
- const bool is_job_id_defined () const
- const bool is_good () const
- const trans_job_code get_code () const
- const string & get_text () const

Static Public Attributes

- static constexpr char HEADER_DELIMITER = ':'
- static constexpr char NEW_LINE_HEADER_ENDING = '\n'

8.94.1 Detailed Description

This class represents the translation reply message, which is a translation result for a translation job. This result can be a text in the target language or it can be an error.

Definition at line 63 of file trans_job_response.hpp.

8.94.2 Constructor & Destructor Documentation

8.94.2.1 uva::smt::bpbd::common::messaging::trans_job_response::trans_job_response ( ) [inline]

The basic no-argument constructor that is needed for the translation client. It default-initializes the class with undefined values.

Definition at line 73 of file trans_job_response.hpp.

8.94.2.2 uva::smt::bpbd::common::messaging::trans_job_response::trans_job_response (const string & *message*) [inline]

This is the basic class constructor that accepts the original server message to parse. This constructor is to be used on the client to de-serialize the resulting message.

Parameters

message	the server message to be parsed

Definition at line 84 of file trans_job_response.hpp.

8.94.2.3 uva::smt::bpbd::common::messaging::trans_job_response::trans_job_response (const job_id_type *job_id,* const trans_job_code *code,* const string & *text*) [inline]

This is the basic class constructor that accepts the translation job id, the translation result code and the text.

Parameters

job_id	the client-issued id of the translation job
code	the translation job result code
text	the translation job result text, either the translated text or the error message corresponding to the error code

Definition at line 99 of file trans_job_response.hpp.

8.94.3 Member Function Documentation

8.94.3.1 void uva::smt::bpbd::common::messaging::trans_job_response::de_serialize (const string & *message*) [inline]

Allows to de-serialize the job reply from a string

Parameters

message	the string representation of the translation job reply

Definition at line 107 of file trans_job_response.hpp.

8.94.3.2 const trans_job_code uva::smt::bpbd::common::messaging::trans_job_response::get_code ( ) const [inline]

Allows to get the translation job result code

Returns

the translation job result code

Definition at line 179 of file trans_job_response.hpp.

8.94.3.3 const job_id_type uva::smt::bpbd::common::messaging::trans_job_response::get_job_id ( ) const [inline]

Allows to get the client-issued job id

Returns

the client-issued job id

Definition at line 154 of file trans_job_response.hpp.

8.94.3.4 const string& uva::smt::bpbd::common::messaging::trans_job_response::get_text ( ) const [inline]

Allows to get the translation job text. This is either the text translated into the target language or the error message for the case of failed translation job request.

Returns

the translation job text

Definition at line 189 of file trans_job_response.hpp.

8.94.3.5 const bool uva::smt::bpbd::common::messaging::trans_job_response::is_good ( ) const [inline]

Allows to check if the reply is good, i.e. contains the translated text and not the error message

Returns

true if the reply is good and contains the translated text.

Definition at line 171 of file trans_job_response.hpp.

8.94.3.6 const bool uva::smt::bpbd::common::messaging::trans_job_response::is_job_id_defined ( ) const [inline]

Allows to check whether the job id is defined, is not equal to job_id::UNDEFINED_JOB_ID;

Returns

true if the job id is defined, otherwise false

Definition at line 163 of file trans_job_response.hpp.

8.94.3.7 const string uva::smt::bpbd::common::messaging::trans_job_response::serialize ( ) [inline]

Allows to serialize the job reply into a string

Returns

the string representation of the translation job reply

Definition at line 141 of file trans_job_response.hpp.

8.94.4 Member Data Documentation

8.94.4.1 constexpr char uva::smt::bpbd::common::messaging::trans_job_response::HEADER_DELIMITER = ':' [static]

Definition at line 66 of file trans_job_response.hpp.

8.94.4.2 constexpr char uva::smt::bpbd::common::messaging::trans_job_response::NEW_LINE_HEADER_ENDING = '\n' [static]

Definition at line 67 of file trans_job_response.hpp.

The documentation for this class was generated from the following file:

- inc/common/messaging/trans_job_response.hpp

8.95 uva::smt::bpbd::client::trans_job_status Class Reference

`#include <trans_job_status.hpp>`

Public Types

- enum values {
  STATUS_UNDEFINED = 0, STATUS_REQ_INITIALIZED = STATUS_UNDEFINED + 1, STATUS_REQ_SENT_GOOD = STATUS_REQ_INITIALIZED + 1, STATUS_REQ_SENT_FAIL = STATUS_REQ_SENT_GOOD + 1,
  STATUS_RES_RECEIVED = STATUS_REQ_SENT_FAIL + 1, size = STATUS_RES_RECEIVED + 1 }

Public Member Functions

- trans_job_status (const values status)
- trans_job_status (const int32_t status_val)
- trans_job_status ()
- void operator= (const values &status)
- bool operator== (const values &status) const
- bool operator< (const values &status) const
- operator string () const
- operator int () const
- const char *const str () const

8.95.1 Detailed Description

This class represents the translation job status. It is to be used on the client. It is needed to trace the client-side translation job status.

Definition at line 48 of file trans_job_status.hpp.

8.95.2 Member Enumeration Documentation

8.95.2.1 enum uva::smt::bpbd::client::trans_job_status::values

Stores the possible status values of the client-side translation job

Enumerator

STATUS_UNDEFINED

STATUS_REQ_INITIALIZED

STATUS_REQ_SENT_GOOD

STATUS_REQ_SENT_FAIL

STATUS_RES_RECEIVED

size

Definition at line 54 of file trans_job_status.hpp.

8.95.3 Constructor & Destructor Documentation

8.95.3.1 uva::smt::bpbd::client::trans_job_status::trans_job_status (const values *status*) [inline]

The basic constructor that allows to initialize the value with the status

Parameters

status	the status value to initialize with

Definition at line 67 of file trans_job_status.hpp.

8.95.3.2 uva::smt::bpbd::client::trans_job_status::trans_job_status (const int32_t *status_val*) [inline]

The basic constructor that allows to initialize the value from an integer

Parameters

status_val	the status value to initialize with

Definition at line 74 of file trans_job_status.hpp.

8.95.3.3 uva::smt::bpbd::client::trans_job_status::trans_job_status ( ) [inline]

The basic constructor that creates an undefined value

Definition at line 86 of file trans_job_status.hpp.

8.95.4 Member Function Documentation

8.95.4.1 uva::smt::bpbd::client::trans_job_status::operator int ( ) const [inline]

The operator allowing to convert the value to an integer

Returns

the the integer value

Definition at line 125 of file trans_job_status.hpp.

8.95.4.2 uva::smt::bpbd::client::trans_job_status::operator string ( ) const [inline]

The operator allowing to convert the value to string

Returns

the string representation of the code

Definition at line 117 of file trans_job_status.hpp.

8.95.4.3 bool uva::smt::bpbd::client::trans_job_status::operator< (const values & *status*) const [inline]

Overloading the comparison operator for the status

Parameters

status	the status to compare with

Definition at line 109 of file trans_job_status.hpp.

8.95.4.4 void uva::smt::bpbd::client::trans_job_status::operator= (const values & *status*) [inline]

Overloading the assignment operator for the status

Parameters

status	the status to set

Definition at line 93 of file trans_job_status.hpp.

8.95.4.5 bool uva::smt::bpbd::client::trans_job_status::operator== (const values & *status*) const [inline]

Overloading the equality operator for the status

Parameters

status	the status to check equality with

Definition at line 101 of file trans_job_status.hpp.

8.95.4.6 const char *const uva::smt::bpbd::client::trans_job_status::str ( ) const

Allows to get the job status string for reporting

Returns

the job status string

Definition at line 49 of file trans_job_status.cpp.

The documentation for this class was generated from the following files:

- inc/client/trans_job_status.hpp
- src/client/trans_job_status.cpp

8.96 uva::smt::bpbd::server::trans_manager Class Reference

`#include <trans_manager.hpp>`

Public Types

- typedef function< void(websocketpp::connection_hdl, trans_job_response &) > response_sender
- typedef std::map< websocketpp::connection_hdl, session_id_type, std::owner_less< websocketpp::connection_hdl > > sessions_map_type
- typedef std::map< session_id_type, websocketpp::connection_hdl > handlers_map_type
- typedef handlers_map_type::iterator handlers_map_iter_type

Public Member Functions

- trans_manager (const size_t num_threads)
- void set_response_sender (response_sender sender)
- virtual ~trans_manager ()
- void set_num_threads (const size_t num_threads)
- void report_run_time_info ()
- void open_session (websocketpp::connection_hdl hdl)
- void translate (websocketpp::connection_hdl hdl, trans_job_request_ptr request_ptr)
- void close_session (websocketpp::connection_hdl hdl)
- void stop ()

Protected Member Functions

- void notify_job_finished (trans_job_ptr trans_job)

8.96.1 Detailed Description

This is a synchronized translation sessions manager class that stores that keeps track of the open translation sessions and their objects.

Definition at line 59 of file trans_manager.hpp.

8.96.2 Member Typedef Documentation

8.96.2.1 typedef handlers_map_type::iterator uva::smt::bpbd::server::trans_manager::handlers_map_iter_type

Definition at line 68 of file trans_manager.hpp.

8.96.2.2 typedef std::map<session_id_type, websocketpp::connection_hdl> uva::smt::bpbd::server::trans_manager::handlers_map_type

Definition at line 67 of file trans_manager.hpp.

8.96.2.3 typedef function<void(websocketpp::connection_hdl, trans_job_response &) > uva::smt::bpbd::server::trans_manager::response_sender

Definition at line 63 of file trans_manager.hpp.

8.96.2.4 typedef std::map<websocketpp::connection_hdl, session_id_type, std::owner_less<websocketpp::connection_hdl> > uva::smt::bpbd::server::trans_manager::sessions_map_type

Definition at line 66 of file trans_manager.hpp.

8.96.3 Constructor & Destructor Documentation

8.96.3.1 uva::smt::bpbd::server::trans_manager::trans_manager (const size_t *num_threads*) [inline]

The basic constructor.

Parameters

num_threads	the number of translation threads to run

Todo {Possibly limit the number of allowed open sessions (from one host and the maximum amount of allowed hosts) This is for later, if the server is put for www access.}

Definition at line 78 of file trans_manager.hpp.

8.96.3.2 virtual uva::smt::bpbd::server::trans_manager::~trans_manager ( ) [inline], [virtual]

The basic destructor

Definition at line 95 of file trans_manager.hpp.

8.96.4 Member Function Documentation

8.96.4.1 void uva::smt::bpbd::server::trans_manager::close_session (websocketpp::connection_hdl *hdl*) [inline]

Allows to erase the session object from the map and return the stored object, synchronized. Returns NULL if there was no session object associated with the given handler.

Parameters

hdl	the connection handler to identify the session object.

Returns

the session object to be removed, is to be deallocated by the caller.

Definition at line 187 of file trans_manager.hpp.

8.96.4.2 void uva::smt::bpbd::server::trans_manager::notify_job_finished (trans_job_ptr *trans_job*) [inline], [protected]

Allows to set the non-error translation result, this will also send the response to the client.

Parameters

trans_job	the pointer to the finished translation job

Definition at line 235 of file trans_manager.hpp.

8.96.4.3 void uva::smt::bpbd::server::trans_manager::open_session (websocketpp::connection_hdl *hdl*) [inline]

Allows to create and register a new session object, synchronized. If for some reason a new session can not be opened, an exception is thrown.

Parameters

hdl	[in] the connection handler to identify the session object.

Definition at line 124 of file trans_manager.hpp.

8.96.4.4 void uva::smt::bpbd::server::trans_manager::report_run_time_info ( ) [inline]

Allows to report the runtime information.

Definition at line 114 of file trans_manager.hpp.

8.96.4.5 void uva::smt::bpbd::server::trans_manager::set_num_threads (const size_t *num_threads*) [inline]

Allows to set the new number of worker threads. This operation should be safe as the new threads are just added to the list and the deleted ones are let to finish their translation task execution.

Parameters

num_threads	the new number of worker threads

Definition at line 107 of file trans_manager.hpp.

8.96.4.6 void uva::smt::bpbd::server::trans_manager::set_response_sender (response_sender *sender*) [inline]

Allows to set the response sender function for sending the replies to the client

Parameters

sender	the s ender functional to be set

Definition at line 88 of file trans_manager.hpp.

8.96.4.7 void uva::smt::bpbd::server::trans_manager::stop ( ) [inline]

Allows to stop the translation manager, i.e. cancel all the jobs and move on.

Definition at line 223 of file trans_manager.hpp.

8.96.4.8 void uva::smt::bpbd::server::trans_manager::translate (websocketpp::connection_hdl *hdl,* trans_job_request_ptr *request_ptr*) [inline]

Allows to schedule a new translation request, synchronized. If there is not session associated with the given connection handler then will through. The scheduled translation job request is from this moment on a responsibility of the underlying object to be managed.

Parameters

hdl	[in] the connection handler to identify the session object.
request_ptr	[in] the translation job request to be stored, not NULL

Definition at line 151 of file trans_manager.hpp.

The documentation for this class was generated from the following file:

- inc/server/trans_manager.hpp

8.97 uva::smt::bpbd::client::trans_manager Class Reference

`#include <trans_manager.hpp>`

Public Types

- typedef vector< trans_job_ptr > jobs_list_type
- typedef jobs_list_type::iterator jobs_list_iter_type
- typedef unordered_map< job_id_type, trans_job_ptr > jobs_map_type
- typedef jobs_map_type::iterator jobs_map_iter_type

Public Member Functions

- trans_manager (const client_config &params)
- virtual ~trans_manager ()
- void start ()
- void wait ()
- void stop ()

Static Public Attributes

- static constexpr uint64_t MIN_SENTENCES_PER_REQUEST = 1

Protected Member Functions

- void write_received_job_result (const uint32_t fis, const uint32_t lis, const trans_job_ptr job, ofstream &target_file)
- void write_result_to_file ()
- void check_jobs_done_and_notify ()
- void set_job_response (trans_job_response *trans_job_resp)
- void notify_conn_closed ()
- void notify_jobs_done ()
- void notify_jobs_sent ()
- void send_translation_jobs ()
- uint64_t get_num_of_sentences ()

8.97.1 Detailed Description

This is the client side translation manager class. It's task is to get the source text from a file and then split it into a number of translation jobs that will be sent to the translation server. The finished translation jobs are collected and once all of them are finished the resulting text is written into the output file.

Definition at line 69 of file trans_manager.hpp.

8.97.2 Member Typedef Documentation

8.97.2.1 typedef jobs_list_type::iterator uva::smt::bpbd::client::trans_manager::jobs_list_iter_type

Definition at line 77 of file trans_manager.hpp.

8.97.2.2 typedef vector<trans_job_ptr> uva::smt::bpbd::client::trans_manager::jobs_list_type

Definition at line 76 of file trans_manager.hpp.

8.97.2.3 typedef jobs_map_type::iterator uva::smt::bpbd::client::trans_manager::jobs_map_iter_type

Definition at line 80 of file trans_manager.hpp.

8.97.2.4 typedef unordered_map<job_id_type, trans_job_ptr> uva::smt::bpbd::client::trans_manager::jobs_↩ map_type

Definition at line 79 of file trans_manager.hpp.

8.97.3 Constructor & Destructor Documentation

8.97.3.1 uva::smt::bpbd::client::trans_manager::trans_manager (const client_config & *params*) [inline]

This is the basic constructor needed to

Parameters

params	the translation client parameters

Definition at line 86 of file trans_manager.hpp.

8.97.3.2 virtual uva::smt::bpbd::client::trans_manager::~trans_manager ( ) [inline], [virtual]

The basic destructor class

Definition at line 126 of file trans_manager.hpp.

8.97.4 Member Function Documentation

8.97.4.1 void uva::smt::bpbd::client::trans_manager::check_jobs_done_and_notify ( ) [inline], [protected]

Allows to check if all the jobs are done and then perform a notifying action

Definition at line 290 of file trans_manager.hpp.

8.97.4.2 uint64_t uva::smt::bpbd::client::trans_manager::get_num_of_sentences ( ) [inline], [protected]

Allows to compute the number of sentences to send with the next request

Returns

the number of sentences to send with the next request

Definition at line 425 of file trans_manager.hpp.

8.97.4.3 void uva::smt::bpbd::client::trans_manager::notify_conn_closed ( ) [inline], [protected]

This function will be called if the connection is closed during the translation process

Definition at line 337 of file trans_manager.hpp.

8.97.4.4 void uva::smt::bpbd::client::trans_manager::notify_jobs_done ( ) [inline], [protected]

Allows to notify the threads waiting on the translation jobs to be received

Definition at line 356 of file trans_manager.hpp.

8.97.4.5 void uva::smt::bpbd::client::trans_manager::notify_jobs_sent ( ) [inline], [protected]

Allows to notify the threads waiting on the translation jobs to be sent

Definition at line 372 of file trans_manager.hpp.

8.97.4.6 void uva::smt::bpbd::client::trans_manager::send_translation_jobs ( ) [inline],[protected]

This function shall be run in a separate thread and send a number of translation job requests to the server.

Definition at line 388 of file trans_manager.hpp.

8.97.4.7 void uva::smt::bpbd::client::trans_manager::set_job_response (trans_job_response * *trans_job_resp*) [inline],[protected]

Allows to process the server job request response

Parameters

trans_job_resp	the translation job response coming from the server

Definition at line 301 of file trans_manager.hpp.

8.97.4.8 void uva::smt::bpbd::client::trans_manager::start ( ) [inline]

Allows to start the translation process

Definition at line 144 of file trans_manager.hpp.

8.97.4.9 void uva::smt::bpbd::client::trans_manager::stop ( ) [inline]

This method allows to stop the translation client and to write the resulting translations into the file.

Definition at line 188 of file trans_manager.hpp.

8.97.4.10 void uva::smt::bpbd::client::trans_manager::wait ( ) [inline]

Allows to wait until the translations are done

Definition at line 156 of file trans_manager.hpp.

8.97.4.11 void uva::smt::bpbd::client::trans_manager::write_received_job_result (const uint32_t *fis,* const uint32_t *lis,* const trans_job_ptr *job,* ofstream & *target_file*) [inline],[protected]

Allows to write the received translation job replies into the file

Parameters

fis	the first sentence number
lis	the last sentence number
job	the translation job data
target_file	the file to write to

Definition at line 215 of file trans_manager.hpp.

8.97.4.12 void uva::smt::bpbd::client::trans_manager::write_result_to_file ( ) [inline],[protected]

Allows to generate the translation result file.

Definition at line 242 of file trans_manager.hpp.

8.97.5 Member Data Documentation

8.97.5.1 constexpr uint64_t uva::smt::bpbd::client::trans_manager::MIN_SENTENCES_PER_REQUEST = 1 [static]

Definition at line 73 of file trans_manager.hpp.

The documentation for this class was generated from the following file:

- inc/client/trans_manager.hpp

8.98 uva::smt::bpbd::server::trans_task Class Reference

`#include <trans_task.hpp>`

Public Types

- typedef function< void(trans_task_ptr) > done_task_notifier
- typedef function< void(trans_task_ptr) > cancel_task_notifier

Public Member Functions

- trans_task (const session_id_type session_id, const job_id_type job_id, const task_id_type task_id, const string &source_sentence, done_task_notifier notify_task_done_func)
- virtual ~trans_task ()
- void set_cancel_task_notifier (cancel_task_notifier notify_task_cancel_func)
- void cancel ()
- void translate ()
- const task_id_type get_task_id () const
- const trans_job_code get_code () const
- const string & get_source_text () const
- const string & get_target_text ()

Protected Member Functions

- void process_task_result ()

8.98.1 Detailed Description

This class represents the translation task. Every translation task is a sentence to be translated and its id.

Definition at line 61 of file trans_task.hpp.

8.98.2 Member Typedef Documentation

8.98.2.1 typedef function<void(trans_task_ptr) > uva::smt::bpbd::server::trans_task::cancel_task_notifier

Definition at line 66 of file trans_task.hpp.

8.98.2.2 typedef function<void(trans_task_ptr) > uva::smt::bpbd::server::trans_task::done_task_notifier

Definition at line 64 of file trans_task.hpp.

8.98.3 Constructor & Destructor Documentation

8.98.3.1 uva::smt::bpbd::server::trans_task::trans_task (const session_id_type *session_id,* const job_id_type *job_id,* const task_id_type *task_id,* const string & *source_sentence,* done_task_notifier *notify_task_done_func*) [inline]

The basic constructor allowing to initialize the main class constants

Parameters

session_id	the session id of the task, is used for logging
job_id	the job id of the task, is used for logging
task_id	the id of the translation task within the translation job
source_sentence	the sentence to be translated

Definition at line 75 of file trans_task.hpp.

8.98.3.2 virtual uva::smt::bpbd::server::trans_task::~trans_task ( ) [inline], [virtual]

The basic destructor

Definition at line 88 of file trans_task.hpp.

8.98.4 Member Function Documentation

8.98.4.1 void uva::smt::bpbd::server::trans_task::cancel ( ) [inline]

Allows to cancel the translation task

Definition at line 104 of file trans_task.hpp.

8.98.4.2 const trans_job_code uva::smt::bpbd::server::trans_task::get_code ( ) const [inline]

Allows to retrieve the translation task result code

Returns

the translation task result code

Definition at line 176 of file trans_task.hpp.

8.98.4.3 const string& uva::smt::bpbd::server::trans_task::get_source_text ( ) const [inline]

Allows to retrieve the sentence in the source language

Returns

the sentence in the source language

Definition at line 184 of file trans_task.hpp.

8.98.4.4 const string& uva::smt::bpbd::server::trans_task::get_target_text ( ) [inline]

Allows to retrieve the sentence in the target language or an error message

Returns

the sentence in the target language or an error message

Definition at line 192 of file trans_task.hpp.

8.98.4.5 const task_id_type uva::smt::bpbd::server::trans_task::get_task_id ( ) const [inline]

Allows to retrieve the task id

Returns

the task id

Definition at line 168 of file trans_task.hpp.

8.98.4.6 void uva::smt::bpbd::server::trans_task::process_task_result ( ) [inline], [protected]

Allows to process the translation task result in case of a successful and abnormal task termination. This includes sending the notification to the translation job that the task is finished. NOTE: This method is not thread safe!

Definition at line 208 of file trans_task.hpp.

8.98.4.7 void uva::smt::bpbd::server::trans_task::set_cancel_task_notifier (cancel_task_notifier *notify_task_cancel_func*) [inline]

Allows to set the function which must be called by the tasks if it is being cancelled.

Parameters

notify_task_↩ cancel_func	the function to call in case this task is being cancelled.

Definition at line 96 of file trans_task.hpp.

8.98.4.8 void uva::smt::bpbd::server::trans_task::translate ( ) [inline]

Performs the translation for the given sentence

Definition at line 126 of file trans_task.hpp.

The documentation for this class was generated from the following file:

- inc/server/trans_task.hpp

8.99 uva::smt::bpbd::server::trans_task_pool Class Reference

`#include <trans_task_pool.hpp>`

Public Types

- typedef deque< trans_task_ptr > tasks_queue_type
- typedef tasks_queue_type::iterator tasks_queue_iter_type
- typedef vector< thread > threads_list_type
- typedef vector< trans_task_pool_worker * > workers_list_type

Public Member Functions

- trans_task_pool (const size_t num_threads)
- void set_num_threads (const size_t num_threads)
- void report_run_time_info ()
- virtual ~trans_task_pool ()
- void plan_new_task (trans_task_ptr trans_task)

Protected Member Functions

- void notify_task_cancel (trans_task_ptr trans_task)

Protected Attributes

- tasks_queue_type m_tasks
- mutex m_queue_mutex
- condition_variable m_condition
- atomic< bool > m_stop

Friends

- class trans_task_pool_worker

8.99.1 Detailed Description

This class represents a translation tasks pool which is in essence a thread pool. One should add the translation tasks into this class and using the pre-configured number of threads it will execute these tasks one by one. This class is thread safe.

Definition at line 32 of file trans_task_pool.hpp.

8.99.2 Member Typedef Documentation

8.99.2.1 typedef tasks_queue_type::iterator uva::smt::bpbd::server::trans_task_pool::tasks_queue_iter_type

Definition at line 37 of file trans_task_pool.hpp.

8.99.2.2 typedef deque<trans_task_ptr> uva::smt::bpbd::server::trans_task_pool::tasks_queue_type

Definition at line 36 of file trans_task_pool.hpp.

8.99.2.3 typedef vector<thread> uva::smt::bpbd::server::trans_task_pool::threads_list_type

Definition at line 40 of file trans_task_pool.hpp.

8.99.2.4 typedef vector<trans_task_pool_worker *> uva::smt::bpbd::server::trans_task_pool::workers_list_type

Definition at line 42 of file trans_task_pool.hpp.

8.99.3 Constructor & Destructor Documentation

8.99.3.1 uva::smt::bpbd::server::trans_task_pool::trans_task_pool (const size_t *num_threads*)

This is a basic constructor accepting the number of threads parameter.

Parameters

num_threads	the number of threads to be run by this task pool.

Definition at line 38 of file trans_task_pool.cpp.

8.99.3.2 uva::smt::bpbd::server::trans_task_pool::~trans_task_pool ( ) [virtual]

The class destructor

Definition at line 108 of file trans_task_pool.cpp.

8.99.4 Member Function Documentation

8.99.4.1 void uva::smt::bpbd::server::trans_task_pool::notify_task_cancel (trans_task_ptr *trans_task*) [protected]

The method that will be called in case a task is canceled

Parameters

trans_task	the task that is being canceled

Todo {To improve performance we could try checking if the tasks is already running, and if not then search the queue. Or use other data structure for a more efficient task removal. This is for the future, in case the performance is affected.}

Definition at line 128 of file trans_task_pool.cpp.

8.99.4.2 void uva::smt::bpbd::server::trans_task_pool::plan_new_task (trans_task_ptr *trans_task*)

This method allows to plan a new translation task

Parameters

trans_task	the translation task to plan

Definition at line 150 of file trans_task_pool.cpp.

8.99.4.3 void uva::smt::bpbd::server::trans_task_pool::report_run_time_info ( ) [inline]

Allows to report the runtime information.

Definition at line 61 of file trans_task_pool.hpp.

8.99.4.4 void uva::smt::bpbd::server::trans_task_pool::set_num_threads (const size_t *new_num_threads*)

Allows to set the new number of worker threads. This operation should be safe as the new threads are just added to the list and the deleted ones are let to finish their translation task execution.

Parameters

num_threads	the new number of worker threads

Allows to set the new number of worker threads

Parameters

new_num_↩ threads	the new number of worker threads

Definition at line 52 of file trans_task_pool.cpp.

8.99.5 Friends And Related Function Documentation

8.99.5.1 friend class trans_task_pool_worker [friend]

Definition at line 93 of file trans_task_pool.hpp.

8.99.6 Member Data Documentation

8.99.6.1 condition_variable uva::smt::bpbd::server::trans_task_pool::m_condition [protected]

Definition at line 100 of file trans_task_pool.hpp.

8.99.6.2 mutex uva::smt::bpbd::server::trans_task_pool::m_queue_mutex [protected]

Definition at line 99 of file trans_task_pool.hpp.

8.99.6.3 atomic<bool> uva::smt::bpbd::server::trans_task_pool::m_stop [protected]

Definition at line 103 of file trans_task_pool.hpp.

8.99.6.4 tasks_queue_type uva::smt::bpbd::server::trans_task_pool::m_tasks [protected]

Definition at line 96 of file trans_task_pool.hpp.

The documentation for this class was generated from the following files:

- inc/server/trans_task_pool.hpp
- src/server/trans_task_pool.cpp

8.100 uva::smt::bpbd::server::trans_task_pool_worker Class Reference

`#include <trans_task_pool_worker.hpp>`

Public Member Functions

- trans_task_pool_worker (trans_task_pool &pool)
- virtual ~trans_task_pool_worker ()
- void stop ()
- atomic< bool > & is_busy ()
- void operator() ()

8.100.1 Detailed Description

This class represents a translation tasks pool worker. This is class is to be used around the actual translation task inside the translation tasks pool. We need this class as a synchronization layer for the thread pool, as each of instances of this class will be run by a thread.

Definition at line 50 of file trans_task_pool_worker.hpp.

8.100.2 Constructor & Destructor Documentation

8.100.2.1 uva::smt::bpbd::server::trans_task_pool_worker::trans_task_pool_worker (trans_task_pool & *pool*) [inline]

This is a basic constructor that needs the thread pool reference as an argument.

Parameters

pool	the task pool reference

Definition at line 57 of file trans_task_pool_worker.hpp.

8.100.2.2 virtual uva::smt::bpbd::server::trans_task_pool_worker::~trans_task_pool_worker ( ) [inline], [virtual]

The basic destructor

Definition at line 64 of file trans_task_pool_worker.hpp.

8.100.3 Member Function Documentation

8.100.3.1 atomic<bool>& uva::smt::bpbd::server::trans_task_pool_worker::is_busy ( ) [inline]

Returns the reference to the worker's is_busy flag

Returns

the worker's is busy flag reference to see if the worker is busy or not.

Definition at line 79 of file trans_task_pool_worker.hpp.

8.100.3.2 void uva::smt::bpbd::server::trans_task_pool_worker::operator() ( )

This operator will be called to run the thread, its implementation will run the tasks scheduled in the thread pool.

Definition at line 34 of file trans_task_pool_worker.cpp.

8.100.3.3 void uva::smt::bpbd::server::trans_task_pool_worker::stop ( ) [inline]

Forces the given worker to stop

Definition at line 71 of file trans_task_pool_worker.hpp.

The documentation for this class was generated from the following files:

- inc/server/trans_task_pool_worker.hpp
- src/server/trans_task_pool_worker.cpp

8.101 uva::smt::bpbd::client::translation_client Class Reference

`#include <translation_client.hpp>`

Public Types

- typedef websocketpp::client< websocketpp::config::asio_client > client
- typedef function< void(const trans_job_response_ptr trans_job_resp) > response_setter
- typedef function< void() > conn_close_notifier

Public Member Functions

- translation_client (const string &host, const uint16_t port, response_setter set_response, conn_close_notifier notify_conn_close)
- ~translation_client ()
- bool connect ()
- void disconnect ()
- void send (const trans_job_request_ptr request)
- void on_message (websocketpp::connection_hdl hdl, client::message_ptr msg)
- void on_open (websocketpp::connection_hdl hdl)
- void on_close (websocketpp::connection_hdl hdl)
- void on_fail (websocketpp::connection_hdl hdl)
- const string get_uri ()

Protected Member Functions

- bool wait_connect ()

8.101.1 Detailed Description

This class is responsible for sending the translation job request to the server and receiving the result.

Definition at line 61 of file translation_client.hpp.

8.101.2 Member Typedef Documentation

8.101.2.1 typedef websocketpp::client<websocketpp::config::asio_client> uva::smt::bpbd::client::translation_client::client

Definition at line 63 of file translation_client.hpp.

8.101.2.2 typedef function<void() > uva::smt::bpbd::client::translation_client::conn_close_notifier

Definition at line 69 of file translation_client.hpp.

8.101.2.3 typedef function<void(const trans_job_response_ptr trans_job_resp) > uva::smt::bpbd::client::translation_client::response_setter

Definition at line 66 of file translation_client.hpp.

8.101.3 Constructor & Destructor Documentation

8.101.3.1 uva::smt::bpbd::client::translation_client::translation_client (const string & *host*, const uint16_t *port*, response_setter *set_response*, conn_close_notifier *notify_conn_close*) [inline]

Definition at line 71 of file translation_client.hpp.

8.101.3.2 uva::smt::bpbd::client::translation_client::~translation_client ( ) [inline]

The basic destructor that also stops the client

Definition at line 97 of file translation_client.hpp.

8.101.4 Member Function Documentation

8.101.4.1 bool uva::smt::bpbd::client::translation_client::connect ( ) [inline]

This method will block until the connection is complete

Parameters

uri	the uri to connect to

Returns

true if the connection has been established

Definition at line 108 of file translation_client.hpp.

8.101.4.2 void uva::smt::bpbd::client::translation_client::disconnect ( ) [inline]

Allows to close the connection and stop the io service thread

Definition at line 134 of file translation_client.hpp.

8.101.4.3 const string uva::smt::bpbd::client::translation_client::get_uri ( ) [inline]

Allows to get the connection URI

Returns

the connection URI

Definition at line 252 of file translation_client.hpp.

8.101.4.4 void uva::smt::bpbd::client::translation_client::on_close (websocketpp::connection_hdl *hdl*) [inline]

The close handler will signal that we should stop sending translation job(s)

Parameters

the	connection handler

Definition at line 218 of file translation_client.hpp.

8.101.4.5 void uva::smt::bpbd::client::translation_client::on_fail (websocketpp::connection_hdl *hdl*) [inline]

The fail handler will signal that we should stop sending translation job(s)

Parameters

the	connection handler

Definition at line 235 of file translation_client.hpp.

8.101.4.6 void uva::smt::bpbd::client::translation_client::on_message (websocketpp::connection_hdl *hdl,* client::message_ptr *msg*) [inline]

This method is used to receive the job translation messages

Parameters

hdl	the connection handler
msg	the message

Definition at line 197 of file translation_client.hpp.

8.101.4.7 void uva::smt::bpbd::client::translation_client::on_open (websocketpp::connection_hdl *hdl*) [inline]

The open handler will signal that we are ready to start sending translation job(s)

Parameters

the	connection handler

Definition at line 206 of file translation_client.hpp.

8.101.4.8 void uva::smt::bpbd::client::translation_client::send (const trans_job_request_ptr *request*) [inline]

Attempts to send the translation job request

Parameters

request	the translation job request

Definition at line 173 of file translation_client.hpp.

8.101.4.9 bool uva::smt::bpbd::client::translation_client::wait_connect ( ) [inline], [protected]

Allows to wait until the connection to the server is established.

Returns

true if the connection is successfully established

Definition at line 262 of file translation_client.hpp.

The documentation for this class was generated from the following file:

- inc/client/translation_client.hpp

8.102 uva::smt::bpbd::server::translation_server Class Reference

`#include <translation_server.hpp>`

Public Types

- typedef websocketpp::server< websocketpp::config::asio > server

Public Member Functions

- translation_server (const uint16_t port, const size_t num_threads)
- void set_num_threads (const size_t num_threads)
- void report_run_time_info ()
- void run ()
- void stop ()

Protected Member Functions

- void send_response (connection_hdl hdl, trans_job_response &response)
- void on_open (connection_hdl hdl)
- void on_close (connection_hdl hdl)
- void on_fail (connection_hdl hdl)
- void on_message (websocketpp::connection_hdl hdl, server::message_ptr msg)

8.102.1 Detailed Description

This is the translation server class implementing the functionality of receiving the client connections and doing translation jobs for them.

Definition at line 64 of file translation_server.hpp.

8.102.2 Member Typedef Documentation

8.102.2.1 typedef websocketpp::server<websocketpp::config::asio> uva::smt::bpbd::server::translation_server::server

Definition at line 66 of file translation_server.hpp.

8.102.3 Constructor & Destructor Documentation

8.102.3.1 uva::smt::bpbd::server::translation_server::translation_server (const uint16_t *port*, const size_t *num_threads*) [inline]

The basic constructor

Parameters

port	the port to listen to
num_threads	the number of translation threads to run

Definition at line 73 of file translation_server.hpp.

8.102.4 Member Function Documentation

8.102.4.1 void uva::smt::bpbd::server::translation_server::on_close (connection_hdl *hdl*) [inline], [protected]

Removes the session object and also stops the processed translation job requests

Parameters

hdl	the connection handler

Definition at line 196 of file translation_server.hpp.

8.102.4.2 void uva::smt::bpbd::server::translation_server::on_fail (connection_hdl *hdl*) [inline], [protected]

Is called in case of a websocket error, for now does nothing but logs the error

Parameters

hdl	the connection handler

Definition at line 207 of file translation_server.hpp.

8.102.4.3 void uva::smt::bpbd::server::translation_server::on_message (websocketpp::connection_hdl *hdl,* server::message_ptr *msg*) [inline], [protected]

Is called when the message is received by the server

Parameters

hdl	the connection handler
msg	the received message

Definition at line 216 of file translation_server.hpp.

8.102.4.4 void uva::smt::bpbd::server::translation_server::on_open (connection_hdl *hdl*) [inline], [protected]

Creates a new session object for the new connection/client

Parameters

hdl	

Definition at line 179 of file translation_server.hpp.

8.102.4.5 void uva::smt::bpbd::server::translation_server::report_run_time_info ( ) [inline]

Allows to report the runtime information about the server.

Definition at line 111 of file translation_server.hpp.

8.102.4.6 void uva::smt::bpbd::server::translation_server::run ( ) [inline]

Allows to run the server

Definition at line 118 of file translation_server.hpp.

8.102.4.7 void uva::smt::bpbd::server::translation_server::send_response (connection_hdl *hdl,* trans_job_response & *response*) [inline], [protected]

Allows to send the translation job response to the client associated with the given connection handler.

Parameters

hdl	the connection handler to identify the connection
response	the translation response object to be used

Definition at line 156 of file translation_server.hpp.

8.102.4.8 void uva::smt::bpbd::server::translation_server::set_num_threads (const size_t *num_threads*) [inline]

Allows to set the new number of worker threads. This operation should be safe as the new threads are just added to the list and the deleted ones are let to finish their translation task execution.

Parameters

num_threads	the new number of worker threads

Definition at line 104 of file translation_server.hpp.

8.102.4.9 void uva::smt::bpbd::server::translation_server::stop ( ) [inline]

Allows to stop the translation server

Definition at line 126 of file translation_server.hpp.

The documentation for this class was generated from the following file:

- inc/server/translation_server.hpp

8.103 uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::TSubArrReference Struct Reference

`#include <c2w_array_trie.hpp>`

Public Attributes

- TShortId begin_idx
- TShortId end_idx

8.103.1 Detailed Description

template<typename WordIndexType>struct uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::TSubArrReference

This structure is needed to store begin and end index to reference pieces of an array It is used to reference sub-array ranges for the M-gram data for levels $1 < M < N$.

WARNING: It is not possible to get rid of this structure as the contexts are not ordered. It is only true that the contexts will be filled one after another, but the context id will not be increased all the time.

Parameters

beginIdx	the begin index
endIdx	the end index

Definition at line 418 of file c2w_array_trie.hpp.

8.103.2 Member Data Documentation

8.103.2.1 template<typename WordIndexType > TShortId uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::TSubArrReference::begin_idx

Definition at line 419 of file c2w_array_trie.hpp.

8.103.2.2 template<typename WordIndexType > TShortId uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::TSubArrReference::end_idx

Definition at line 420 of file c2w_array_trie.hpp.

The documentation for this struct was generated from the following file:

- inc/server/lm/models/c2w_array_trie.hpp

8.104 uva::smt::bpbd::server::lm::__C2WArrayTrie::TWordIdPBData Struct Reference

`#include <c2w_array_trie.hpp>`

Public Attributes

- TShortId id
- m_gram_payload payload

8.104.1 Detailed Description

This structure stores two things the word id and the corresponding probability/back-off data. It is used to store the M-gram data for levels $1 < M < N$.

Parameters

id	the word id
payload	the back-off and probability data

Definition at line 58 of file c2w_array_trie.hpp.

8.104.2 Member Data Documentation

8.104.2.1 TShortId uva::smt::bpbd::server::lm::__C2WArrayTrie::TWordIdPBData::id

Definition at line 59 of file c2w_array_trie.hpp.

8.104.2.2 m_gram_payload uva::smt::bpbd::server::lm::__C2WArrayTrie::TWordIdPBData::payload

Definition at line 60 of file c2w_array_trie.hpp.

The documentation for this struct was generated from the following file:

- inc/server/lm/models/c2w_array_trie.hpp

8.105 uva::smt::bpbd::server::lm::dictionary::__counting_word_index::TWordInfo Struct Reference

`#include <counting_word_index.hpp>`

Public Attributes

- string word
- prob_weight prob

8.105.1 Detailed Description

Definition at line 56 of file counting_word_index.hpp.

8.105.2 Member Data Documentation

8.105.2.1 prob_weight uva::smt::bpbd::server::lm::dictionary::__counting_word_index::TWordInfo::prob

Definition at line 58 of file counting_word_index.hpp.

8.105.2.2 string uva::smt::bpbd::server::lm::dictionary::__counting_word_index::TWordInfo::word

Definition at line 57 of file counting_word_index.hpp.

The documentation for this struct was generated from the following file:

- inc/server/lm/dictionaries/counting_word_index.hpp

8.106 uva::utils::containers::upp_diag_matrix< element_type > Class Template Reference

`#include <upp_diag_matrix.hpp>`

Public Types

- typedef element_type * element_type_ptr

Public Member Functions

- upp_diag_matrix (const size_t dim)
- ~upp_diag_matrix ()
- const size_t & get_dim () const
- element_type * operator[ ] (size_t idx) const

Public Attributes

- const int32_t m_max_idx

Static Public Attributes

- static constexpr int32_t m_min_idx = 0

8.106.1 Detailed Description

template<typename element_type>class uva::utils::containers::upp_diag_matrix< element_type >

This class represents the square matrix that should save on memory when used with e.g. upper/low diagonal matrix rows. Note 1: This class is thread safe as long as you do not try to work with the same element from different threads. Note 2: This class assumes the proper acces to the matrix elements. I.e. matrix indexes are not checked, writing the the elements below the diagonal will also result in broken data.

Definition at line 56 of file upp_diag_matrix.hpp.

8.106.2 Member Typedef Documentation

8.106.2.1 template<typename element_type> typedef element_type* uva::utils::containers::upp_diag_matrix< element_type >::element_type_ptr

Definition at line 59 of file upp_diag_matrix.hpp.

8.106.3 Constructor & Destructor Documentation

8.106.3.1 template<typename element_type> uva::utils::containers::upp_diag_matrix< element_type >::upp_diag_matrix (const size_t *dim*) [inline]

The basic constructor

Parameters

dimension	the dimension of the matrix, it will be a square upper diagonal matrix.

Definition at line 71 of file upp_diag_matrix.hpp.

8.106.3.2 template<typename element_type> uva::utils::containers::upp_diag_matrix< element_type >::~upp_diag_matrix ( ) [inline]

The basic destructor

Definition at line 99 of file upp_diag_matrix.hpp.

8.106.4 Member Function Documentation

8.106.4.1 template<typename element_type> const size_t& uva::utils::containers::upp_diag_matrix< element_type >::get_dim ( ) const [inline]

Allows to retrieve the reference to the dimension of this square matrix

Returns

the reference to the dimension of the scquare matrix

Definition at line 116 of file upp_diag_matrix.hpp.

8.106.4.2 template<typename element_type> element_type* uva::utils::containers::upp_diag_matrix< element_type >::operator[ ] (size_t *idx*) const [inline]

Allows to access the matrix row with the given index

Parameters

idx	the row index

Returns

the pointer to the matrix row array

Definition at line 125 of file upp_diag_matrix.hpp.

8.106.5 Member Data Documentation

8.106.5.1 template<typename element_type> const int32_t uva::utils::containers::upp_diag_matrix< element_type >::m_max_idx

Definition at line 65 of file upp_diag_matrix.hpp.

8.106.5.2 template<typename element_type> constexpr int32_t uva::utils::containers::upp_diag_matrix< element_type >::m_min_idx = 0 [static]

Definition at line 62 of file upp_diag_matrix.hpp.

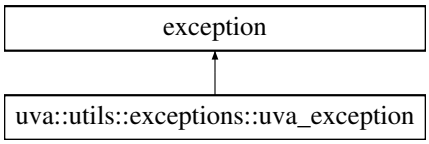
The documentation for this class was generated from the following file:

- inc/common/utils/containers/upp_diag_matrix.hpp

8.107 uva::utils::exceptions::uva_exception Class Reference

`#include <exceptions.hpp>`

Inheritance diagram for uva::utils::exceptions::uva_exception:



Public Member Functions

- uva_exception (const char *message)
- uva_exception (const string &message)
- uva_exception (uva_exception const &other)
- string const & get_message () const throw ()
- virtual ~uva_exception () throw ()
- virtual const char * what () const throw ()

8.107.1 Detailed Description

This is an application exception class that is capable of storing an error message

Definition at line 69 of file exceptions.hpp.

8.107.2 Constructor & Destructor Documentation

8.107.2.1 uva::utils::exceptions::uva_exception::uva_exception (const char * *message*) [inline], [explicit]

Definition at line 76 of file exceptions.hpp.

8.107.2.2 uva::utils::exceptions::uva_exception::uva_exception (const string & *message*) [inline], [explicit]

Definition at line 79 of file exceptions.hpp.

8.107.2.3 uva::utils::exceptions::uva_exception::uva_exception (uva_exception const & *other*) [inline]

The copy constructor

Parameters

other	the other exception to copy from

Definition at line 86 of file exceptions.hpp.

8.107.2.4 virtual uva::utils::exceptions::uva_exception::~uva_exception ( ) throw) [inline], [virtual]

Destructor. Virtual to allow for subclassing.

Definition at line 101 of file exceptions.hpp.

8.107.3 Member Function Documentation

8.107.3.1 string const& uva::utils::exceptions::uva_exception::get_message ( ) const throw) [inline]

This method returns the stored message

Returns

the reference to a constant error message string

Definition at line 94 of file exceptions.hpp.

8.107.3.2 virtual const char* uva::utils::exceptions::uva_exception::what ( ) const throw) [inline], [virtual]

Returns a pointer to the (constant) error description.

Returns

A pointer to a `const char*`. The underlying memory is in posession of the `Exception` object. Callers *must* not attempt to free the memory.

Definition at line 109 of file exceptions.hpp.

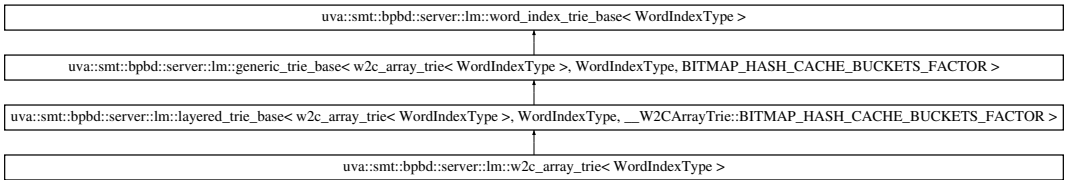
The documentation for this class was generated from the following file:

- inc/common/utils/exceptions.hpp

8.108 uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType > Class Template Reference

`#include <w2c_array_trie.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >:



Classes

- class WordDataEntry

Public Types

- typedef layered_trie_base< w2c_array_trie< WordIndexType >, WordIndexType, __W2CArrayTrie::BITM↩ AP_HASH_CACHE_BUCKETS_FACTOR > BASE

Public Member Functions

- w2c_array_trie (WordIndexType &word_index)
- float get_unk_word_prob () const
- bool get_ctx_id (const phrase_length level_idx, const TShortId word_id, TLongId &ctx_id) const
- void log_model_type_info () const
- void set_def_unk_word_prob (const prob_weight prob)
- virtual void pre_allocate (const size_t counts[LM_M_GRAM_LEVEL_MAX])
- template<phrase_length CURR_LEVEL>
  bool is_post_grams () const
- template<phrase_length CURR_LEVEL>
  void post_grams ()
- template<phrase_length CURR_LEVEL>
  void add_m_gram (const model_m_gram &gram)
- void get_unigram_payload (m_gram_query &query) const
- void get_m_gram_payload (m_gram_query &query, MGramStatusEnum &status) const
- void get_n_gram_payload (m_gram_query &query, MGramStatusEnum &status) const
- virtual ~w2c_array_trie ()

Protected Types

- typedef WordDataEntry< T_M_GramData > T_M_GramWordEntry
- typedef WordDataEntry< T_N_GramData > T_N_GramWordEntry

Protected Member Functions

- template<typename WORD_ENTRY_TYPE >
  void post_M_N_Grams (WORD_ENTRY_TYPE *wordsArray)
- template<phrase_length level>
  void post_m_grams ()
- void post_n_grams ()

Additional Inherited Members

8.108.1 Detailed Description

template<typename WordIndexType>class uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >

This is the Context to word array memory trie implementation class.

Parameters

M_GRAM_LEVEL_MAX	the maximum number of levels in the trie.

Definition at line 107 of file w2c_array_trie.hpp.

8.108.2 Member Typedef Documentation

8.108.2.1 template<typename WordIndexType > typedef layered_trie_base<w2c_array_trie<WordIndexType>, WordIndexType, __W2CArrayTrie::BITMAP_HASH_CACHE_BUCKETS_FACTOR> uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::BASE

Definition at line 109 of file w2c_array_trie.hpp.

8.108.2.2 template<typename WordIndexType > typedef WordDataEntry<T_M_GramData> uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::T_M_GramWordEntry [protected]

Definition at line 382 of file w2c_array_trie.hpp.

8.108.2.3 template<typename WordIndexType > typedef WordDataEntry<T_N_GramData> uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::T_N_GramWordEntry [protected]

Definition at line 384 of file w2c_array_trie.hpp.

8.108.3 Constructor & Destructor Documentation

8.108.3.1 template<typename WordIndexType > uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::w2c_array_trie (WordIndexType & *word_index*) [explicit]

The basic constructor

Parameters

p_word_index	the word index (dictionary) container

Definition at line 48 of file w2c_array_trie.cpp.

8.108.3.2 template<typename WordIndexType > uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::~w2c_array_trie ( ) [virtual]

The basic destructor

Definition at line 89 of file w2c_array_trie.cpp.

8.108.4 Member Function Documentation

8.108.4.1 template<typename WordIndexType > template<phrase_length CURR_LEVEL> void uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::add_m_gram (const model_m_gram & *gram*) [inline]

Allows to retrieve the data storage structure for the M gram with the given M-gram level Id. M-gram context and last word Id. If the storage structure does not exist, return a new one. For more details

See also

LayeredTrieBase

Definition at line 241 of file w2c_array_trie.hpp.

8.108.4.2 template<typename WordIndexType > bool uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::get_ctx_id (const phrase_length *level_idx,* const TShortId *word_id,* TLongId & *ctx_id*) const [inline]

Computes the M-Gram context using the previous context and the current word id

See also

LayeredTrieBese

Definition at line 129 of file w2c_array_trie.hpp.

8.108.4.3 template<typename WordIndexType > void uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::get_m_gram_payload (m_gram_query & *query,* MGramStatusEnum & *status*) const [inline]

Allows to retrieve the payload for the M-gram defined by the end word_id and ctx_id. For more details

See also

LayeredTrieBase

Definition at line 294 of file w2c_array_trie.hpp.

8.108.4.4 template<typename WordIndexType > void uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::get_n_gram_payload (m_gram_query & *query,* MGramStatusEnum & *status*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for m==n

See also

GenericTrieBase

Definition at line 330 of file w2c_array_trie.hpp.

8.108.4.5 template<typename WordIndexType > void uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::get_unigram_payload (m_gram_query & *query*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for m==1. The retrieval of a uni-gram data is always a success

See also

GenericTrieBase

Definition at line 279 of file w2c_array_trie.hpp.

8.108.4.6 template<typename WordIndexType > float uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::get_unk_word_prob ( ) const [inline]

Allows to retrieve the unknown target word log probability penalty

Returns

the target source word log probability penalty

Definition at line 121 of file w2c_array_trie.hpp.

8.108.4.7 template<typename WordIndexType > template<phrase_length CURR_LEVEL> bool uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::is_post_grams ( ) const [inline]

This method allows to check if post processing should be called after all the X level grams are read. This method is virtual. For more details

See also

WordIndexTrieBase

Definition at line 204 of file w2c_array_trie.hpp.

8.108.4.8 template<typename WordIndexType > void uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::log_model_type_info ( ) const [inline]

Allows to log the information about the instantiated trie type

Definition at line 180 of file w2c_array_trie.hpp.

8.108.4.9 template<typename WordIndexType > template<phrase_length CURR_LEVEL> void uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::post_grams ( ) [inline]

This method should be called after all the X level grams are read. For more details

See also

WordIndexTrieBase

Definition at line 218 of file w2c_array_trie.hpp.

8.108.4.10 template<typename WordIndexType > template<phrase_length level> void uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::post_m_grams ( ) [inline], [protected]

Definition at line 421 of file w2c_array_trie.hpp.

8.108.4.11 template<typename WordIndexType > template<typename WORD_ENTRY_TYPE > void uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::post_M_N_Grams (WORD_ENTRY_TYPE * *wordsArray*) [inline], [protected]

The purpose of this local function is three fold:

1. First we compute the context index offset values.
2. Second we re-order the context data arrays per word.
3. Free the unneeded memory allocated earlier.

Parameters

WORD_ENTRY_TYPE	word array element type
wordsArray	the word array to work with

Definition at line 395 of file w2c_array_trie.hpp.

8.108.4.12 template<typename WordIndexType > void uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::post_n_grams ( ) [inline], [protected]

Definition at line 426 of file w2c_array_trie.hpp.

8.108.4.13 template<typename WordIndexType > void uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::pre_allocate (const size_t *counts[LM_M_GRAM_LEVEL_MAX]*) [virtual]

This method can be used to provide the N-gram count information That should allow for pre-allocation of the memory For more details

See also

LayeredTrieBase

Definition at line 61 of file w2c_array_trie.cpp.

8.108.4.14 template<typename WordIndexType > void uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::set_def_unk_word_prob (const prob_weight *prob*)

See also

word_index_trie_base

Definition at line 81 of file w2c_array_trie.cpp.

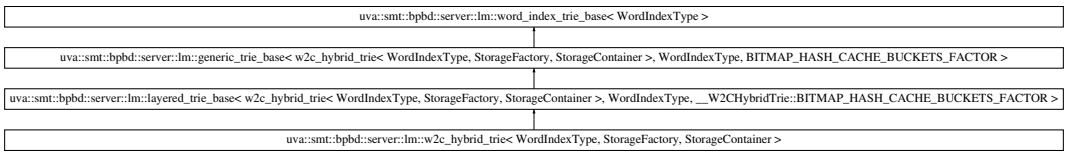
The documentation for this class was generated from the following files:

- inc/server/lm/models/w2c_array_trie.hpp
- src/server/lm/models/w2c_array_trie.cpp

8.109 uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer > Class Template Reference

`#include <w2c_hybrid_trie.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, Storage↩ Container >:



Public Types

- typedef layered_trie_base< w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer >, WordIndexType, __W2CHybridTrie::BITMAP_HASH_CACHE_BUCKETS_FACTOR > BASE

Public Member Functions

- w2c_hybrid_trie (WordIndexType &word_index)
- float get_unk_word_prob () const
- bool get_ctx_id (const phrase_length level_idx, const TShortId word_id, TLongId &ctx_id) const
- void log_model_type_info () const
- void set_def_unk_word_prob (const prob_weight prob)
- virtual void pre_allocate (const size_t counts[LM_M_GRAM_LEVEL_MAX])
- template<phrase_length CURR_LEVEL>
  void add_m_gram (const model_m_gram &gram)
- void get_unigram_payload (m_gram_query &query) const
- void get_m_gram_payload (m_gram_query &query, MGramStatusEnum &status) const
- void get_n_gram_payload (m_gram_query &query, MGramStatusEnum &status) const
- virtual ~w2c_hybrid_trie ()

Additional Inherited Members

8.109.1 Detailed Description

template<typename WordIndexType, template< phrase_length > class StorageFactory = W2CH_UM_StorageFactory, class StorageContainer = W2CH_UM_Storage>class uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer >

This is the hybrid memory trie implementation class. It has three template parameters.

Parameters

M_GRAM_LE↩ VEL_MAX	the maximum number of levelns in the trie.
StorageFactory	the factory to create storage containers
Storage↩ Container	the storage container type that is created by the factory

Definition at line 56 of file w2c_hybrid_trie.hpp.

8.109.2 Member Typedef Documentation

8.109.2.1 template<typename WordIndexType , template< phrase_length > class StorageFactory = W2CH_UM_StorageFactory, class StorageContainer = W2CH_UM_Storage> typedef layered_trie_base<w2c_hybrid_trie<WordIndexType, StorageFactory, StorageContainer>, WordIndexType, __W2CHybridTrie::BITMAP_HASH_CACHE_BUCKETS_FACTOR> uva::smt↩ ::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer >::BASE

Definition at line 58 of file w2c_hybrid_trie.hpp.

8.109.3 Constructor & Destructor Documentation

8.109.3.1 template<typename WordIndexType , template< phrase_length > class StorageFactory, class StorageContainer > uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer >::w2c_hybrid_trie (WordIndexType & *word_index*) [explicit]

The basic constructor

Parameters

p_word_index	the word index (dictionary) container

Definition at line 47 of file w2c_hybrid_trie.cpp.

8.109.3.2 template<typename WordIndexType , template< phrase_length > class StorageFactory, class StorageContainer > uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer >::~w2c_hybrid_trie ( ) [virtual]

The basic destructor

Definition at line 116 of file w2c_hybrid_trie.cpp.

8.109.4 Member Function Documentation

8.109.4.1 template<typename WordIndexType , template< phrase_length > class StorageFactory = W2CH_UM_StorageFactory, class StorageContainer = W2CH_UM_Storage> template<phrase_length CURR_LEVEL> void uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer >::add_m_gram (const model_m_gram & *gram*) [inline]

Allows to retrieve the data storage structure for the M gram with the given M-gram level Id. M-gram context and last word Id. If the storage structure does not exist, return a new one. For more details

See also

LayeredTrieBase

Definition at line 135 of file w2c_hybrid_trie.hpp.

8.109.4.2 template<typename WordIndexType , template< phrase_length > class StorageFactory = W2CH_UM_StorageFactory, class StorageContainer = W2CH_UM_Storage> bool uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer >::get_ctx_id (const phrase_length *level_idx,* const TShortId *word_id,* TLongId & *ctx_id*) const [inline]

Computes the M-Gram context using the previous context and the current word id

See also

LayeredTrieBese

Definition at line 78 of file w2c_hybrid_trie.hpp.

8.109.4.3 template<typename WordIndexType , template< phrase_length > class StorageFactory = W2CH_UM_StorageFactory, class StorageContainer = W2CH_UM_Storage> void uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer >::get_m_gram_payload (m_gram_query & *query,* MGramStatusEnum & *status*) const [inline]

Allows to retrieve the payload for the M-gram defined by the end word_id and ctx_id. For more details

See also

LayeredTrieBase

Definition at line 203 of file w2c_hybrid_trie.hpp.

8.109.4.4 template<typename WordIndexType , template< phrase_length > class StorageFactory = W2CH_UM_StorageFactory, class StorageContainer = W2CH_UM_Storage> void uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer >::get_n_gram_payload (m_gram_query & *query,* MGramStatusEnum & *status*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for m==n

See also

GenericTrieBase

Definition at line 241 of file w2c_hybrid_trie.hpp.

8.109.4.5 template<typename WordIndexType , template< phrase_length > class StorageFactory = W2CH_UM_StorageFactory, class StorageContainer = W2CH_UM_Storage> void uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer >::get_unigram_payload (m_gram_query & *query*) const [inline]

Allows to attempt the sub-m-gram payload retrieval for m==1. The retrieval of a uni-gram data is always a success

See also

GenericTrieBase

Definition at line 188 of file w2c_hybrid_trie.hpp.

8.109.4.6 template<typename WordIndexType , template< phrase_length > class StorageFactory = W2CH_UM_StorageFactory, class StorageContainer = W2CH_UM_Storage> float uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer >::get_unk_word_prob ( ) const [inline]

Allows to retrieve the unknown target word log probability penalty

Returns

the target source word log probability penalty

Definition at line 70 of file w2c_hybrid_trie.hpp.

8.109.4.7 template<typename WordIndexType , template< phrase_length > class StorageFactory = W2CH_UM_StorageFactory, class StorageContainer = W2CH_UM_Storage> void uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer >::log_model_type_info ( ) const [inline]

Allows to log the information about the instantiated trie type

Definition at line 112 of file w2c_hybrid_trie.hpp.

8.109.4.8 template<typename WordIndexType , template< phrase_length > class StorageFactory, class StorageContainer > void uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer >::pre_allocate (const size_t *counts[LM_M_GRAM_LEVEL_MAX]*) [virtual]

This method can be used to provide the N-gram count information That should allow for pre-allocation of the memory For more details

See also

LayeredTrieBase

Definition at line 72 of file w2c_hybrid_trie.cpp.

8.109.4.9 template<typename WordIndexType , template< phrase_length > class StorageFactory, class StorageContainer > void uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer >::set_def_unk_word_prob (const prob_weight *prob*)

See also

word_index_trie_base

Definition at line 107 of file w2c_hybrid_trie.cpp.

The documentation for this class was generated from the following files:

- inc/server/lm/models/w2c_hybrid_trie.hpp
- src/server/lm/models/w2c_hybrid_trie.cpp

8.110 uva::smt::bpbd::server::lm::W2CH_UM_Storage Class Reference

`#include <w2ch_um_storage.hpp>`

Public Types

- typedef TStorageUnsignedMap::const_iterator const_iterator

Public Member Functions

- W2CH_UM_Storage (TStorageMapAllocator &alloc)
- virtual ~W2CH_UM_Storage ()
- TShortId & operator[ ] (const TShortId ctx_idx)
- const TShortId & at (const TShortId ctx_idx) const throw (out_of_range)
- const_iterator find (const TShortId ctx_idx)
- const_iterator end ()

8.110.1 Detailed Description

The unordered hash map-based storage for the HybridMemoryTrie

Definition at line 61 of file w2ch_um_storage.hpp.

8.110.2 Member Typedef Documentation

8.110.2.1 typedef TStorageUnsignedMap::const_iterator uva::smt::bpbd::server::lm::W2CH_UM_Storage::const_iterator

Definition at line 64 of file w2ch_um_storage.hpp.

8.110.3 Constructor & Destructor Documentation

8.110.3.1 uva::smt::bpbd::server::lm::W2CH_UM_Storage::W2CH_UM_Storage (TStorageMapAllocator & *alloc*) [inline]

Definition at line 66 of file w2ch_um_storage.hpp.

8.110.3.2 virtual uva::smt::bpbd::server::lm::W2CH_UM_Storage::~W2CH_UM_Storage ( ) [inline], [virtual]

Definition at line 70 of file w2ch_um_storage.hpp.

8.110.4 Member Function Documentation

8.110.4.1 const TShortId& uva::smt::bpbd::server::lm::W2CH_UM_Storage::at (const TShortId *ctx_idx*) const throw out_of_range) [inline]

Definition at line 78 of file w2ch_um_storage.hpp.

8.110.4.2 const_iterator uva::smt::bpbd::server::lm::W2CH_UM_Storage::end ( ) [inline]

Definition at line 86 of file w2ch_um_storage.hpp.

8.110.4.3 const_iterator uva::smt::bpbd::server::lm::W2CH_UM_Storage::find (const TShortId *ctx_idx*) [inline]

Definition at line 82 of file w2ch_um_storage.hpp.

8.110.4.4 TShortId& uva::smt::bpbd::server::lm::W2CH_UM_Storage::operator[] (const TShortId *ctx_idx*) [inline]

Definition at line 74 of file w2ch_um_storage.hpp.

The documentation for this class was generated from the following file:

- inc/server/lm/models/w2ch_um_storage.hpp

8.111 uva::smt::bpbd::server::lm::W2CH_UM_StorageFactory< N > Class Template Reference

`#include <w2ch_um_storage.hpp>`

Public Member Functions

- W2CH_UM_StorageFactory (const size_t _counts[N], const float factor=__W2CHybridTrie::UM_CTX_TO_↩ PB_MAP_STORE_MEMORY_FACTOR)
- virtual ~W2CH_UM_StorageFactory ()
- W2CH_UM_Storage * create (const phrase_length level)

Protected Attributes

- TStorageMapAllocator * m_p_alloc [N-1]

8.111.1 Detailed Description

template<phrase_length N>class uva::smt::bpbd::server::lm::W2CH_UM_StorageFactory< N >

This is a factory class that should be used to produce containers of CtxToPBMapStorage.

Definition at line 99 of file w2ch_um_storage.hpp.

8.111.2 Constructor & Destructor Documentation

8.111.2.1 template<phrase_length N> uva::smt::bpbd::server::lm::W2CH_UM_StorageFactory< N >::W2CH_UM_StorageFactory (const size_t *_counts[N],* const float *factor =* __W2CHybridTrie::UM_CTX_TO_PB_MAP_STORE_MEMORY_FACTOR) [inline]

This is a basic constructor for the factory

Parameters

_counts	the number of elements to insert per trie level
factor	the memory multiplication factor, by default __CtxToPBMapStorageFactory::UM_CTX_TO↩ _PB_MAP_STORE_MEMORY_FACTOR

Definition at line 108 of file w2ch_um_storage.hpp.

8.111.2.2 template<phrase_length N> virtual uva::smt::bpbd::server::lm::W2CH_UM_StorageFactory< N >::~W2CH_UM_StorageFactory ( ) [inline], [virtual]

The basic destructor

Definition at line 122 of file w2ch_um_storage.hpp.

8.111.3 Member Function Documentation

8.111.3.1 template<phrase_length N> W2CH_UM_Storage* uva::smt::bpbd::server↩ ::lm::W2CH_UM_StorageFactory< N >::create (const phrase_length *level*) [inline]

Allocates a new storage container for the given M-gram level

Parameters

level	the N-gram level must be > 1 and <= N

Returns

the pointer to the allocated container

Definition at line 133 of file w2ch_um_storage.hpp.

8.111.4 Member Data Documentation

8.111.4.1 template<phrase_length N> TStorageMapAllocator* uva::smt::bpbd::server::lm::W2CH_UM_StorageFactory< N >::m_p_alloc[N-1] [protected]

Definition at line 144 of file w2ch_um_storage.hpp.

The documentation for this class was generated from the following file:

- inc/server/lm/models/w2ch_um_storage.hpp

8.112 uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index::word_index_bucket_entry< word_id_type > Struct Template Reference

`#include <optimizing_word_index.hpp>`

Public Member Functions

- word_index_bucket_entry ()

Public Attributes

- char * m_word
- uint8_t m_len
- word_id_type m_word_id

8.112.1 Detailed Description

template<typename word_id_type>struct uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index::word_index_bucket_entry< word_id_type >

This structure is to store the word index data, the word itself and its index

Definition at line 65 of file optimizing_word_index.hpp.

8.112.2 Constructor & Destructor Documentation

8.112.2.1 template<typename word_id_type > uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index::word_index_bucket_entry< word_id_type >::word_index_bucket_entry ( ) [inline]

Definition at line 67 of file optimizing_word_index.hpp.

8.112.3 Member Data Documentation

8.112.3.1 template<typename word_id_type > uint8_t uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index::word_index_bucket_entry< word_id_type >::m_len

Definition at line 70 of file optimizing_word_index.hpp.

8.112.3.2 template<typename word_id_type > char* uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index::word_index_bucket_entry< word_id_type >::m_word

Definition at line 69 of file optimizing_word_index.hpp.

8.112.3.3 template<typename word_id_type > word_id_type uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index::word_index_bucket_entry< word_id_type >::m_word_id

Definition at line 71 of file optimizing_word_index.hpp.

The documentation for this struct was generated from the following file:

- inc/server/lm/dictionaries/optimizing_word_index.hpp

8.113 uva::smt::bpbd::server::lm::word_index_trie_base< WordIndex > Class Template Reference

`#include <word_index_trie_base.hpp>`

Public Types

- typedef WordIndex WordIndexType

Public Member Functions

- word_index_trie_base (WordIndexType &word_index)
- void pre_allocate (const size_t counts[LM_M_GRAM_LEVEL_MAX])
- template<phrase_length level>
  bool is_post_grams () const
- template<phrase_length level>
  void post_grams ()
- void set_def_unk_word_prob (const prob_weight prob)
- WordIndexType & get_word_index () const

Protected Attributes

- WordIndexType & m_word_index

8.113.1 Detailed Description

template<typename WordIndex>class uva::smt::bpbd::server::lm::word_index_trie_base< WordIndex >

This is a common base class for all Trie implementations. The purpose of having this as a template class is performance optimization.

Definition at line 59 of file word_index_trie_base.hpp.

8.113.2 Member Typedef Documentation

8.113.2.1 template<typename WordIndex> typedef WordIndex uva::smt::bpbd::server::lm::word_index_trie_base< WordIndex >::WordIndexType

Definition at line 61 of file word_index_trie_base.hpp.

8.113.3 Constructor & Destructor Documentation

8.113.3.1 template<typename WordIndex> uva::smt::bpbd::server::lm::word_index_trie_base< WordIndex >::word_index_trie_base (WordIndexType & *word_index*) [inline], [explicit]

The basic constructor

Parameters

word_index	the word index to be used

Definition at line 67 of file word_index_trie_base.hpp.

8.113.4 Member Function Documentation

8.113.4.1 template<typename WordIndex> WordIndexType& uva::smt::bpbd::server::lm::word_index_trie_base< WordIndex >::get_word_index ( ) const [inline]

Allows to retrieve the stored word index, if any

Returns

the pointer to the stored word index or NULL if none

Definition at line 112 of file word_index_trie_base.hpp.

8.113.4.2 template<typename WordIndex> template<phrase_length level> bool uva::smt::bpbd::server::lm::word_index_trie_base< WordIndex >::is_post_grams ( ) const [inline]

This method allows to check if post processing should be called after all the X level grams are read. This method is virtual.

Parameters

level	the level of the X-grams that were finished to be read

Definition at line 87 of file word_index_trie_base.hpp.

8.113.4.3 template<typename WordIndex> template<phrase_length level> void uva::smt::bpbd::server::lm::word_index_trie_base< WordIndex >::post_grams ( ) [inline]

This method should be called after all the X level grams are read.

Parameters

level	the level of the X-grams that were finished to be read

Definition at line 96 of file word_index_trie_base.hpp.

8.113.4.4 template<typename WordIndex> void uva::smt::bpbd::server::lm::word_index_trie_base< WordIndex >::pre_allocate (const size_t *counts[LM_M_GRAM_LEVEL_MAX]*) [inline]

This method can be used to provide the N-gram count information That should allow for pre-allocation of the memory

Parameters

counts	the array of N-Gram counts counts[0] is for 1-Gram

Definition at line 76 of file word_index_trie_base.hpp.

8.113.4.5 template<typename WordIndex> void uva::smt::bpbd::server::lm::word_index_trie_base< WordIndex >::set_def_unk_word_prob (const prob_weight *prob*) [inline]

Allows to set the default UNK word probability value, the back-off is set to zero

Parameters

prob	the unk word default probability value

Definition at line 104 of file word_index_trie_base.hpp.

8.113.5 Member Data Documentation

8.113.5.1 template<typename WordIndex> WordIndexType& uva::smt::bpbd::server::lm::word_index_trie_base< WordIndex >::m_word_index [protected]

Definition at line 114 of file word_index_trie_base.hpp.

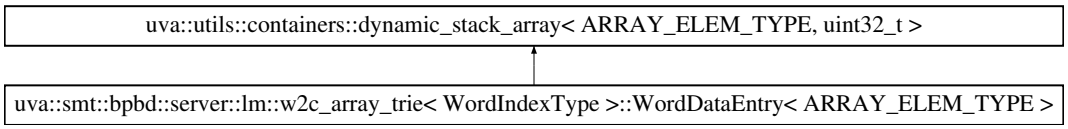
The documentation for this class was generated from the following file:

- inc/server/lm/models/word_index_trie_base.hpp

8.114 uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::WordDataEntry< ARRAY_ELEM_TYPE > Class Template Reference

`#include <w2c_array_trie.hpp>`

Inheritance diagram for uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::WordDataEntry< ARRAY_ELEM_TYPE >:



Public Attributes

- TShortId cio

Additional Inherited Members

8.114.1 Detailed Description

template<typename WordIndexType>template<typename ARRAY_ELEM_TYPE>class uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::WordDataEntry< ARRAY_ELEM_TYPE >

This class is to store the word mapping to the data for the 1< M <= N grams. Demending on whether M == N or not this structure is to be instantiated with a different template parameter - defines the stored data.

Parameters

ptr	the pointer to the storage array
capacity	the number of allocated elements
size	the number of used elements
cio	the context index offset for computing the next contex index.

Definition at line 376 of file w2c_array_trie.hpp.

8.114.2 Member Data Documentation

8.114.2.1 template<typename WordIndexType > template<typename ARRAY_ELEM_TYPE > TShortId uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::WordDataEntry< ARRAY_ELEM_TYPE >::cio

Definition at line 378 of file w2c_array_trie.hpp.

The documentation for this class was generated from the following file:

- inc/server/lm/models/w2c_array_trie.hpp

Chapter 9

File Documentation

9.1 inc/client/client_config.hpp File Reference

Classes

- struct uva::smt::bpbd::client::client_config

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::client

9.2 inc/client/trans_job.hpp File Reference

```
#include "trans_job_status.hpp"
#include "common/utils/exceptions.hpp"
#include "common/messaging/trans_job_request.hpp"
#include "common/messaging/trans_job_response.hpp"
```

Classes

- struct uva::smt::bpbd::client::trans_job

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::client

Typedefs

- typedef trans_job * uva::smt::bpbd::client::trans_job_ptr

9.3 inc/server/trans_job.hpp File Reference

```
#include <string>
#include <vector>
#include "trans_task.hpp"
#include "common/utils/threads.hpp"
#include "common/messaging/id_manager.hpp"
#include "common/messaging/trans_session_id.hpp"
#include "common/messaging/trans_job_request.hpp"
#include "common/messaging/trans_job_id.hpp"
#include "common/messaging/trans_job_code.hpp"
```

Classes

- class uva::smt::bpbd::server::trans_job

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server

Typedefs

- typedef trans_job * uva::smt::bpbd::server::trans_job_ptr

9.4 inc/client/trans_job_status.hpp File Reference

```
#include <string>
#include <iostream>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
```

Classes

- class uva::smt::bpbd::client::trans_job_status

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::client

Functions

- ostream & uva::smt::bpbd::client::operator<< (ostream &os, const trans_job_status &status)

9.5 inc/client/trans_manager.hpp File Reference

```
#include <string>
#include <vector>
#include <unordered_map>
#include <cstdlib>
#include <chrono>
#include <iostream>
#include <fstream>
#include "client_config.hpp"
#include "translation_client.hpp"
#include "trans_job.hpp"
#include "trans_job_status.hpp"
#include "common/messaging/trans_job_code.hpp"
#include "common/messaging/id_manager.hpp"
#include "common/messaging/trans_job_id.hpp"
#include "common/messaging/trans_job_request.hpp"
#include "common/messaging/trans_job_response.hpp"
#include "common/utils/file/cstyle_file_reader.hpp"
#include "common/utils/threads.hpp"
#include "common/utils/string_utils.hpp"
```

Classes

- class uva::smt::bpbd::client::trans_manager

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::client

9.6 inc/server/trans_manager.hpp File Reference

```
#include <map>
#include <websocketpp/server.hpp>
#include "common/utils/threads.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/messaging/trans_job_request.hpp"
#include "common/messaging/id_manager.hpp"
#include "common/messaging/trans_session_id.hpp"
#include "common/messaging/trans_job_id.hpp"
#include "trans_job_pool.hpp"
#include "trans_job.hpp"
```

Classes

- class uva::smt::bpbd::server::trans_manager

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server

9.7 inc/client/translation_client.hpp File Reference

```
#include <cstdlib>
#include <string>
#include <unordered_map>
#include <websocketpp/config/asio_no_tls_client.hpp>
#include <websocketpp/client.hpp>
#include "common/utils/threads.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/messaging/trans_job_response.hpp"
#include "common/messaging/trans_job_request.hpp"
```

Classes

- class uva::smt::bpbd::client::translation_client

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::client

Macros

- #define ASIO_STANDALONE

9.7.1 Macro Definition Documentation

9.7.1.1 #define ASIO_STANDALONE

Definition at line 33 of file translation_client.hpp.

9.8 inc/common/messaging/id_manager.hpp File Reference

```
#include <websocketpp/common/thread.hpp>
```

Classes

- class uva::smt::bpbd::common::messaging::id_manager< id_type >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::common
- uva::smt::bpbd::common::messaging

9.9 inc/common/messaging/trans_job_code.hpp File Reference

```
#include <string>
#include <iostream>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
```

Classes

- class uva::smt::bpbd::common::messaging::trans_job_code

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::common
- uva::smt::bpbd::common::messaging

Functions

- ostream & uva::smt::bpbd::common::messaging::operator<< (ostream &os, const trans_job_code &code)

9.10 inc/common/messaging/trans_job_id.hpp File Reference

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::common
- uva::smt::bpbd::common::messaging
- uva::smt::bpbd::common::messaging::job_id

Typedefs

- typedef uint64_t uva::smt::bpbd::common::messaging::job_id_type

9.11 inc/common/messaging/trans_job_request.hpp File Reference

```
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/file/text_piece_reader.hpp"
#include "common/messaging/trans_session_id.hpp"
#include "common/messaging/trans_job_id.hpp"
```

Classes

- class uva::smt::bpbd::common::messaging::trans_job_request

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::common
- uva::smt::bpbd::common::messaging

Typedefs

- typedef trans_job_request * uva::smt::bpbd::common::messaging::trans_job_request_ptr

9.12 inc/common/messaging/trans_job_response.hpp File Reference

```
#include <string>
#include <sstream>
#include <iostream>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/file/text_piece_reader.hpp"
#include "common/messaging/trans_job_id.hpp"
#include "common/messaging/trans_job_request.hpp"
#include "common/messaging/trans_job_code.hpp"
```

Classes

- class uva::smt::bpbd::common::messaging::trans_job_response

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::common
- uva::smt::bpbd::common::messaging

Typedefs

- typedef trans_job_response * uva::smt::bpbd::common::messaging::trans_job_response_ptr

9.13 inc/common/messaging/trans_session_id.hpp File Reference

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::common
- uva::smt::bpbd::common::messaging
- uva::smt::bpbd::common::messaging::session_id

Typedefs

- typedef uint64_t uva::smt::bpbd::common::messaging::session_id_type

9.14 inc/common/utils/containers/array_utils.hpp File Reference

```
#include <string>
#include <cstdlib>
#include <algorithm>
#include <functional>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
```

Classes

- struct uva::utils::containers::utils::T_IS_COMPARE_FUNC< ELEM_TYPE >

Namespaces

- uva
- uva::utils
- uva::utils::containers
- uva::utils::containers::utils

Macros

- #define BSEARCH_ONE_FIELD(FIELD_NAME, RETURN_STATEMENT)
- #define BSEARCH_TWO_FIELDS(FIELD_ONE, FIELD_TWO)
- #define DECLARE_STATIC_BSEARCH_ID_FIELD_COMPARE_FUNC(COMPARE_STATEMENT, ...)

Functions

- template<typename ARR_ELEM_TYPE >
  bool uva::utils::containers::utils::my_bsearch_id (const ARR_ELEM_TYPE *array, int64_t l_idx, int64_t u_idx, const typename ARR_ELEM_TYPE::TIdType key, const ARR_ELEM_TYPE *&found_elem)
- template<typename ARR_ELEM_TYPE , typename IDX_TYPE , typename KEY_TYPE >
  bool uva::utils::containers::utils::my_bsearch_id (const ARR_ELEM_TYPE *array, int64_t l_idx, int64_t u_idx, const KEY_TYPE key, IDX_TYPE &found_pos)
- template<typename ARR_ELEM_TYPE >
  bool uva::utils::containers::utils::my_bsearch_wordId_ctxId (const ARR_ELEM_TYPE *array, int64_t l_idx, int64_t u_idx, const uint32_t key1, const uint32_t key2, uint32_t &found_pos)
- template<typename ARR_ELEM_TYPE , typename KEY_TYPE >
  bool uva::utils::containers::utils::my_isearch_id (const ARR_ELEM_TYPE *array, int64_t l_idx, int64_t u_idx, const KEY_TYPE key, const ARR_ELEM_TYPE *&found_elem)
- template<typename ARR_ELEM_TYPE , typename INDEX_TYPE , typename KEY_TYPE >
  bool uva::utils::containers::utils::my_bsearch (const ARR_ELEM_TYPE *array, INDEX_TYPE l_idx, INDE↩X_TYPE u_idx, const KEY_TYPE key, INDEX_TYPE &mid_pos)
- template<typename ARR_ELEM_TYPE >
  bool uva::utils::containers::utils::my_lsearch_id (const ARR_ELEM_TYPE *array, int64_t l_idx, int64_t u_idx, const typename ARR_ELEM_TYPE::TIdType key, const ARR_ELEM_TYPE *&found_elem)
- template<typename ELEM_TYPE >
  void uva::utils::containers::utils::my_sort (ELEM_TYPE *array_begin, const uint32_t array_size, typename T_IS_COMPARE_FUNC< ELEM_TYPE >::func_type is_less_func)
- template<typename ELEM_TYPE , typename T_IS_COMPARE_FUNC< ELEM_TYPE >::func_ptr IS_LESS_FUNC>
  void uva::utils::containers::utils::my_sort (ELEM_TYPE *array_begin, const uint32_t array_size)
- template<typename ELEM_TYPE , bool IS_PROGRESS = true>
  bool uva::utils::containers::utils::is_less (const ELEM_TYPE &first, const ELEM_TYPE &second)

9.14.1 Macro Definition Documentation

9.14.1.1 #define BSEARCH_ONE_FIELD(*FIELD_NAME, RETURN_STATEMENT*)

Value:

```
ASSERT_SANITY_THROW(((l_idx < 0) || (l_idx > u_idx)), \
                        string("Impossible search parameters, l_idx = ") + \
                        std::to_string(l_idx) + string(", u_idx = ") + \
                        std::to_string(u_idx) + string("!")); \
                int64_t mid_pos;                                                    \
                while (l_idx <= u_idx) {                                            \
                    mid_pos = (l_idx + u_idx) / 2;                                  \
                    LOG_DEBUG4 << "l_idx = " << SSTR(l_idx) << ", u_idx = "         \
                            << SSTR(u_idx) << ", mid_pos = " << SSTR(mid_pos) <<
      END_LOG;   \
                    if (key < array[mid_pos].FIELD_NAME) {                          \
                        u_idx = mid_pos - 1;                                        \
                    } else {                                                        \
                        if (key > array[mid_pos].FIELD_NAME) {                      \
                            l_idx = mid_pos + 1;                                    \
                        } else {                                                    \
                            LOG_DEBUG4 << "The found mid_pos = "                    \
                                    << SSTR(mid_pos) << END_LOG;                    \
                            RETURN_STATEMENT;                                       \
                            return true;                                            \
                        }                                                           \
                    }                                                               \
                }                                                                   \
                return false;
```

Definition at line 60 of file array_utils.hpp.

9.14.1.2 #define BSEARCH_TWO_FIELDS(*FIELD_ONE, FIELD_TWO*)

Definition at line 90 of file array_utils.hpp.

9.14.1.3 #define DECLARE_STATIC_BSEARCH_ID_FIELD_COMPARE_FUNC(*COMPARE_STATEMENT, ...*)

Value:

```
template<typename ARR_ELEM_TYPE> \
            static inline bool my_bsearch_id(const ARR_ELEM_TYPE * array, \
                    int64_t l_idx, int64_t u_idx, \
                    const ARR_ELEM_TYPE * & found_elem, __VA_ARGS__) { \
                ASSERT_SANITY_THROW(((l_idx < 0) || (l_idx > u_idx)), \
                        string("Impossible search parameters, l_idx = ") + \
                        std::to_string(l_idx) + string(", u_idx = ") + \
                        std::to_string(u_idx) + string("!")); \
                int64_t mid_pos; \
                while (l_idx <= u_idx) { \
                    mid_pos = (l_idx + u_idx) / 2; \
                    LOG_DEBUG4 << "l_idx = " << SSTR(l_idx) << ", u_idx = " \
                            << SSTR(u_idx) << ", mid_pos = " << SSTR(mid_pos) <<
      END_LOG; \
                    int64_t result = COMPARE_STATEMENT; \
                    if (result < 0) { \
                        u_idx = mid_pos - 1; \
                    } else { \
                        if (result == 0) { \
                            LOG_DEBUG4 << "The found mid_pos = " << SSTR(mid_pos) <<
      END_LOG; \
                            found_elem = &array[mid_pos]; \
                            return true; \
                        } else { \
                            l_idx = mid_pos + 1; \
                        } \
                    } \
                } \
                return false; \
            }
```

This is a binary search algorithm for some ordered array

Parameters

ARR_ELEM_TYPE	the array element structure, must have id field as this method will specifically use it to compare elements.
COMPARE_STATEMENT	the compare statement that is to return a compare result
array	the pointer to the first array element
l_idx	the initial left border index for searching
u_idx	the initial right border index for searching
found_elem	the out parameter that stores the pointer to the found element, if any
the	variable list of arguments needed for the compare statement

Returns

true if the element was found, otherwise false

Exceptions

Exception	in case (l_idx $<$ 0) $||$ (l_idx $>$ u_idx), with sanity checks on

Definition at line 136 of file array_utils.hpp.

9.15 inc/common/utils/containers/circular_queue.hpp File Reference

```
#include <string>
#include <ostream>
#include <cstring>
#include <algorithm>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/string_utils.hpp"
```

Classes

- class uva::utils::containers::circular_queue< elem_type, capacity >

Namespaces

- uva
- uva::utils
- uva::utils::containers

9.16 inc/common/utils/containers/dynamic_memory_arrays.hpp File Reference

```
#include <functional>
#include <cmath>
#include <algorithm>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/containers/array_utils.hpp"
#include "common/utils/math_utils.hpp"
```

Classes

- class uva::utils::containers::mem_increase_strategy
- struct uva::utils::containers::ELEMENT_DEALLOC_FUNC< ELEM_TYPE >
- class uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >

Namespaces

- uva
- uva::utils
- uva::utils::containers

Macros

- #define EXTRACT_P(NAME_PTR) ELEMENT_TYPE_PTR & NAME_PTR = extract_bytes<sizeof (IDX_DATA_TYPE), ELEMENT_TYPE_PTR > (m_params);
- #define EXTRACT_C(NAME_CAPACITY) IDX_DATA_TYPE & NAME_CAPACITY = extract_bytes<0, IDX_DATA_TYPE > (m_params);
- #define EXTRACT_S(NAME_SIZE) IDX_DATA_TYPE & NAME_SIZE = extract_bytes<sizeof (ELEMENT_TYPE_PTR) + sizeof (IDX_DATA_TYPE), IDX_DATA_TYPE > (m_params);
- #define EXTRACT_PC(NAME_PTR, NAME_CAPACITY)
- #define EXTRACT_PS(NAME_PTR, NAME_SIZE)
- #define EXTRACT_PCS(NAME_PTR, NAME_CAPACITY, NAME_SIZE)

Typedefs

- typedef std::function< size_t(const size_t) > uva::utils::containers::TCapacityIncFunct

Enumerations

- enum uva::utils::containers::mem_inc_types_enum { uva::utils::containers::UNDEFINED = 0, uva::utils::containers::CONSTANT = UNDEFINED + 1, uva::utils::containers::LINEAR = CONSTANT + 1, uva::utils::containers::LOG_2 = LINEAR + 1, uva::utils::containers::LOG_10 = LOG_2 + 1, uva::utils::containers::size = LOG_10 + 1 }

Functions

- mem_increase_strategy uva::utils::containers::get_mem_incr_strat (const mem_inc_types_enum stype, const size_t min_mem_inc, const size_t mem_inc_factor)

Variables

- const char *const uva::utils::containers::_memIncTypesEnumStr [mem_inc_types_enum::size] = {"CONSTANT", "LINEAR", "LOG_2", "LOG_10"}

9.16.1 Macro Definition Documentation

9.16.1.1 #define EXTRACT_C(*NAME_CAPACITY*) IDX_DATA_TYPE & NAME_CAPACITY = extract_bytes<0, IDX_DATA_TYPE > (m_params);

Definition at line 236 of file dynamic_memory_arrays.hpp.

9.16.1.2 #define EXTRACT_P(*NAME_PTR*) ELEMENT_TYPE_PTR & NAME_PTR = extract_bytes<sizeof (IDX_DATA_TYPE), ELEMENT_TYPE_PTR > (m_params);

Definition at line 233 of file dynamic_memory_arrays.hpp.

9.16.1.3 #define EXTRACT_PC(*NAME_PTR, NAME_CAPACITY*)

Value:

```
EXTRACT_P(NAME_PTR);                \
        EXTRACT_C(NAME_CAPACITY);
```

Definition at line 242 of file dynamic_memory_arrays.hpp.

9.16.1.4 #define EXTRACT_PCS(*NAME_PTR, NAME_CAPACITY, NAME_SIZE*)

Value:

```
EXTRACT_P(NAME_PTR);        \
                              EXTRACT_C(NAME_CAPACITY);   \
                              EXTRACT_S(NAME_SIZE);
```

Definition at line 250 of file dynamic_memory_arrays.hpp.

9.16.1.5 #define EXTRACT_PS(*NAME_PTR, NAME_SIZE*)

Value:

```
EXTRACT_P(NAME_PTR);            \
        EXTRACT_S(NAME_SIZE);
```

Definition at line 246 of file dynamic_memory_arrays.hpp.

9.16.1.6 #define EXTRACT_S(*NAME_SIZE*) IDX_DATA_TYPE & NAME_SIZE = extract_bytes<sizeof (ELEMENT_TYPE_PTR) + sizeof (IDX_DATA_TYPE), IDX_DATA_TYPE > (m_params);

Definition at line 239 of file dynamic_memory_arrays.hpp.

9.17 inc/common/utils/containers/fixed_size_hashmap.hpp File Reference

```
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/math_utils.hpp"
#include "common/utils/hashing_utils.hpp"
#include "common/utils/containers/array_utils.hpp"
```

Classes

- class uva::utils::containers::fixed_size_hashmap< ELEMENT_TYPE, KEY_TYPE, IDX_TYPE >

Namespaces

- uva
- uva::utils
- uva::utils::containers

9.18 inc/common/utils/containers/greedy_memory_allocator.hpp File Reference

```
#include <typeinfo>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/containers/greedy_memory_storage.hpp"
```

Classes

- class uva::utils::containers::alloc::greedy_memory_allocator< T >
- struct uva::utils::containers::alloc::greedy_memory_allocator< T >::rebind< U >

Namespaces

- uva
- uva::utils
- uva::utils::containers
- uva::utils::containers::alloc

Functions

- template<typename TContaner , typename TAllocator >
  void uva::utils::containers::alloc::allocate_container (TContaner **ppContainer, TAllocator **ppAllocator, const size_t numEntries, const string ctName, const float factor=UNORDERED_MAP_MEMORY_FACT↩ OR)

- template<typename TContaner , typename TAllocator >
  void uva::utils::containers::alloc::reserve_mem_unordered_map (TContaner **ppContainer, TAllocator **ppAllocator, const size_t numEntries, const string ctName, const float factor=UNORDERED_MAP_↩ MEMORY_FACTOR)
- template<typename TContaner , typename TAllocator >
  void uva::utils::containers::alloc::deallocate_container (TContaner **ppContainer, TAllocator **ppAllocator)
- template<typename T , typename U >
  bool uva::utils::containers::alloc::operator== (const greedy_memory_allocator< T > &, const greedy_↩ memory_allocator< U > &)
- template<typename T >
  bool uva::utils::containers::alloc::operator== (const greedy_memory_allocator< T > &, const greedy_↩ memory_allocator< T > &)
- template<typename T , typename U >
  bool uva::utils::containers::alloc::operator!= (const greedy_memory_allocator< T > &, const greedy_↩ memory_allocator< U > &)
- template<typename T >
  bool uva::utils::containers::alloc::operator!= (const greedy_memory_allocator< T > &, const greedy_↩ memory_allocator< T > &)

9.19 inc/common/utils/containers/greedy_memory_storage.hpp File Reference

```
#include <vector>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
```

Classes

- class uva::utils::containers::greedy_memory_storage

Namespaces

- uva
- uva::utils
- uva::utils::containers

9.20 inc/common/utils/containers/upp_diag_matrix.hpp File Reference

```
#include <string>
#include <cstdlib>
#include <algorithm>
#include <functional>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
```

Classes

- class uva::utils::containers::upp_diag_matrix< element_type >

Namespaces

- uva
- uva::utils
- uva::utils::containers

9.21 inc/common/utils/exceptions.hpp File Reference

```
#include <exception>
#include <string>
#include "common/utils/logging/logger.hpp"
```

Classes

- class uva::utils::exceptions::uva_exception

Namespaces

- uva
- uva::utils
- uva::utils::exceptions

Macros

- #define THROW_EXCEPTION(text)
- #define THROW_MUST_OVERRIDE() THROW_EXCEPTION("Must be overridden in the sub class!")
- #define THROW_MUST_NOT_CALL() THROW_EXCEPTION("Must not be called, is not needed!")
- #define THROW_NOT_IMPLEMENTED() THROW_EXCEPTION("This functionality is not yet implemented!")
- #define ASSERT_CONDITION_THROW(CONDITION, MESSAGE)
- #define ASSERT_SANITY_THROW(CONDITION, MESSAGE) ASSERT_CONDITION_THROW(DO_SA↩ NITY_CHECKS && (CONDITION), MESSAGE);

Variables

- constexpr bool uva::utils::exceptions::DO_SANITY_CHECKS = false

9.21.1 Macro Definition Documentation

9.21.1.1 #define ASSERT_CONDITION_THROW(*CONDITION, MESSAGE*)

Value:

```
if(CONDITION) { \
        THROW_EXCEPTION(MESSAGE); \
    }
```

Definition at line 59 of file exceptions.hpp.

9.21.1.2 #define ASSERT_SANITY_THROW(*CONDITION, MESSAGE*) ASSERT_CONDITION_THROW(DO_SANITY_C↩ HECKS && (CONDITION), MESSAGE);

Definition at line 63 of file exceptions.hpp.

9.21.1.3 #define THROW_EXCEPTION(*text*)

Value:

```
{ \
    stringstream msg; \
    if (logger::get_reporting_level() >= debug_levels_enum::INFO3) { \
        msg << __FILENAME__ << "::" << __FUNCTION__ \
            << "(...) " << __LINE__ << " : " << (text); \
    } else { \
        msg << __FILENAME__ << ": " << (text); \
    } \
    LOG_DEBUG << "<THROWING> " << msg.str() << END_LOG; \
    throw uva_exception(msg.str()); \
}
```

Definition at line 44 of file exceptions.hpp.

9.21.1.4 #define THROW_MUST_NOT_CALL( ) THROW_EXCEPTION("Must not be called, is not needed!")

Definition at line 57 of file exceptions.hpp.

9.21.1.5 #define THROW_MUST_OVERRIDE( ) THROW_EXCEPTION("Must be overridden in the sub class!")

Definition at line 56 of file exceptions.hpp.

9.21.1.6 #define THROW_NOT_IMPLEMENTED( ) THROW_EXCEPTION("This functionality is not yet implemented!")

Definition at line 58 of file exceptions.hpp.

9.22 inc/common/utils/file/afile_reader.hpp File Reference

```
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/string_utils.hpp"
#include "common/utils/file/text_piece_reader.hpp"
```

Classes

- class uva::utils::file::afile_reader

Namespaces

- uva
- uva::utils
- uva::utils::file

9.23 inc/common/utils/file/cstyle_file_reader.hpp File Reference

```
#include <cstring>
```

```
#include <cstdio>
#include <stdio.h>
#include <cstdlib>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/string_utils.hpp"
#include "common/utils/file/afile_reader.hpp"
```

Classes

- class uva::utils::file::cstyle_file_reader

Namespaces

- uva
- uva::utils
- uva::utils::file

9.24 inc/common/utils/file/file_stream_reader.hpp File Reference

```
#include <cstring>
#include <fstream>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/string_utils.hpp"
#include "common/utils/file/text_piece_reader.hpp"
```

Classes

- class uva::utils::file::file_stream_reader

Namespaces

- uva
- uva::utils
- uva::utils::file

9.25 inc/common/utils/file/memory_mapped_file_reader.hpp File Reference

```
#include <string>
```

```
#include <unistd.h>
#include <fcntl.h>
#include <stdio.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <inttypes.h>
#include <stdlib.h>
#include <sys/mman.h>
#include <stdint.h>
#include <cstring>
#include <errno.h>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/string_utils.hpp"
#include "common/utils/file/text_piece_reader.hpp"
#include "common/utils/file/afile_reader.hpp"
```

Classes

- class uva::utils::file::memory_mapped_file_reader

Namespaces

- uva
- uva::utils
- uva::utils::file

9.26 inc/common/utils/file/text_piece_reader.hpp File Reference

```
#include <string.h>
#include <cstring>
#include <algorithm>
#include <string>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/string_utils.hpp"
```

Classes

- class uva::utils::file::text_piece_reader

Namespaces

- uva
- uva::utils
- uva::utils::file

Functions

- ostream & uva::utils::file::operator<< (ostream &output, const text_piece_reader &val)

- template<size_t NUM_TOKENS>
  string uva::utils::file::tokens_to_string (const text_piece_reader tokens[NUM_TOKENS], const size_t begin↩
  _idx, const size_t end_idx)

9.27 inc/common/utils/hashing_utils.hpp File Reference

```
#include <string>
#include <cmath>
#include <stdint.h>
#include "common/utils/logging/logger.hpp"
```

Namespaces

- uva
- uva::utils
- uva::utils::hashing

Macros

- #define get16bits(d)
- #define A 54059 /∗ a prime ∗/
- #define B 76963 /∗ another prime ∗/
- #define C 86969 /∗ yet another prime ∗/
- #define cwfold(a, b, lo, hi) { p = (uint32_t)(a) ∗ (uint64_t)(b); lo ^= (uint32_t)p; hi ^= (uint32_t)(p >> 32); }
- #define cwmixa(in) { cwfold(in, m, k, h); }
- #define cwmixb(in) { cwfold(in, n, h, k); }

9.27.1 Macro Definition Documentation

9.27.1.1 #define A 54059 /∗ a prime ∗/

This is a hash function found online http://stackoverflow.com/questions/8317508/hash-function-for-a-st
It's origin is unknown but it proves to work perfect (without collisions) on both test sets! So I do not need to complicate a hash map to a multi-map for now! Note: The time complexity is linear in the length of the word. Note: There are no observed collisions up until now. Note: But it is not yet known if this hash is collision free.

Parameters

str	the word to hash

Returns

the resulting hash

Definition at line 217 of file hashing_utils.hpp.

9.27.1.2 #define B 76963 /∗ another prime ∗/

Definition at line 218 of file hashing_utils.hpp.

9.27.1.3 #define C 86969 /∗ yet another prime ∗/

Definition at line 219 of file hashing_utils.hpp.

9.27.1.4 #define cwfold(*a, b, lo, hi*) { p = (uint32_t)(a) * (uint64_t)(b); lo ^= (uint32_t)p; hi ^= (uint32_t)(p >> 32); }

9.27.1.5 #define cwmixa(*in*) { cwfold(in, m, k, h); }

9.27.1.6 #define cwmixb(*in*) { cwfold(in, n, h, k); }

9.27.1.7 #define get16bits(*d*)

Value:

```
((((uint_fast32_t)(((const uint8_t *)(d))[1])) << 8)\
                      +(uint_fast32_t)(((const uint8_t *)(d))[0]) )
```

The string hashing functions: computePaulHsiehHash - This one showed the worst speed on a test run compute↩ Djb2Hash - This one showed medium speed on a test run computePrimesHash - This one showed medium speed on a test run computeRSHash - This one showed the best speed on a test run

Note that the XXHASH should be the best with respect to everything, see: https://github.com/↩ Cyan4973/xxHash

Yet it is not even on a 64 bit machine with XXH64 it is beated by RSHash! At least the hash based trie performs faster (200 vs 250 CPU seconds) on a 20 Gb model with 100.000.000 queries. So for us XXHASH is not the best. The following is the Paul Hsieh implementation of a string hashing function This one seems to be very efficient in computation time and has good distribution: http://www.azillionmonkeys.com/qed/hash.html

Definition at line 118 of file hashing_utils.hpp.

9.28 inc/common/utils/logging/logger.hpp File Reference

```
#include <mutex>
#include <iostream>
#include <sstream>
#include <vector>
#include <time.h>
#include <string.h>
```

Classes

- struct uva::utils::logging::logging_synch
- class uva::utils::logging::logger

Namespaces

- uva
- uva::utils
- uva::utils::logging

Macros

- #define SSTR(x) std::dec << (x)
- #define STRINGIZE(x) STRINGIZE2(x)
- #define STRINGIZE2(x) #x
- #define LINE_STRING STRINGIZE(__LINE__)
- #define PROGRESS_UPDATE_PERIOD 0.05

- #define LOGGER(level)
- #define __FILENAME__ (strrchr(__FILE__, '/') ? strrchr(__FILE__, '/') + 1 : __FILE__)
- #define LOGGER_DEBUG(level)
- #define LOG_ERROR LOGGER(debug_levels_enum::ERROR)
- #define LOG_WARNING LOGGER(debug_levels_enum::WARNING)
- #define LOG_USAGE LOGGER(debug_levels_enum::USAGE)
- #define LOG_RESULT LOGGER(debug_levels_enum::RESULT)
- #define LOG_INFO LOGGER(debug_levels_enum::INFO)
- #define LOG_INFO1 LOGGER(debug_levels_enum::INFO1)
- #define LOG_INFO2 LOGGER(debug_levels_enum::INFO2)
- #define LOG_INFO3 LOGGER(debug_levels_enum::INFO3)
- #define LOG_DEBUG LOGGER_DEBUG(debug_levels_enum::DEBUG)
- #define LOG_DEBUG1 LOGGER_DEBUG(debug_levels_enum::DEBUG1)
- #define LOG_DEBUG2 LOGGER_DEBUG(debug_levels_enum::DEBUG2)
- #define LOG_DEBUG3 LOGGER_DEBUG(debug_levels_enum::DEBUG3)
- #define LOG_DEBUG4 LOGGER_DEBUG(debug_levels_enum::DEBUG4)
- #define END_LOG
- #define ERROR_PARAM_VALUE "ERROR"
- #define WARNING_PARAM_VALUE "WARN"
- #define USAGE_PARAM_VALUE "USAGE"
- #define RESULT_PARAM_VALUE "RESULT"
- #define INFO_PARAM_VALUE "INFO"
- #define INFO1_PARAM_VALUE "INFO1"
- #define INFO2_PARAM_VALUE "INFO2"
- #define INFO3_PARAM_VALUE "INFO3"
- #define DEBUG_PARAM_VALUE "DEBUG"
- #define DEBUG1_PARAM_VALUE "DEBUG1"
- #define DEBUG2_PARAM_VALUE "DEBUG2"
- #define DEBUG3_PARAM_VALUE "DEBUG3"
- #define DEBUG4_PARAM_VALUE "DEBUG4"
- #define WHITE_SPACE_SEPARATOR " "

Enumerations

- enum uva::utils::logging::debug_levels_enum {
  uva::utils::logging::ERROR = 0, uva::utils::logging::WARNING = ERROR + 1, uva::utils::logging::USAGE = WARNING + 1, uva::utils::logging::RESULT = USAGE + 1,
  uva::utils::logging::INFO = RESULT + 1, uva::utils::logging::INFO1 = INFO + 1, uva::utils::logging::INFO2 = INFO1 + 1, uva::utils::logging::INFO3 = INFO2 + 1,
  uva::utils::logging::DEBUG = INFO3 + 1, uva::utils::logging::DEBUG1 = DEBUG + 1, uva::utils::logging::D↩
  EBUG2 = DEBUG1 + 1, uva::utils::logging::DEBUG3 = DEBUG2 + 1,
  uva::utils::logging::DEBUG4 = DEBUG3 + 1, uva::utils::logging::size = DEBUG4 + 1 }

Functions

- std::ostream & uva::utils::logging::operator<< (std::ostream &stream, const unsigned char &value)
- std::ostream & uva::utils::logging::operator<< (std::ostream &stream, const signed char &value)

9.28.1 Macro Definition Documentation

9.28.1.1 #define __FILENAME__ (strrchr(__FILE__, '/') ? strrchr(__FILE__, '/') + 1 : __FILE__)

Definition at line 90 of file logger.hpp.

9.28.1.2 #define DEBUG1_PARAM_VALUE "DEBUG1"

Definition at line 131 of file logger.hpp.

9.28.1.3 #define DEBUG2_PARAM_VALUE "DEBUG2"

Definition at line 132 of file logger.hpp.

9.28.1.4 #define DEBUG3_PARAM_VALUE "DEBUG3"

Definition at line 133 of file logger.hpp.

9.28.1.5 #define DEBUG4_PARAM_VALUE "DEBUG4"

Definition at line 134 of file logger.hpp.

9.28.1.6 #define DEBUG_PARAM_VALUE "DEBUG"

Definition at line 130 of file logger.hpp.

9.28.1.7 #define END_LOG

Value:

```
endl << flush; \
}
```

Definition at line 117 of file logger.hpp.

9.28.1.8 #define ERROR_PARAM_VALUE "ERROR"

Definition at line 122 of file logger.hpp.

9.28.1.9 #define INFO1_PARAM_VALUE "INFO1"

Definition at line 127 of file logger.hpp.

9.28.1.10 #define INFO2_PARAM_VALUE "INFO2"

Definition at line 128 of file logger.hpp.

9.28.1.11 #define INFO3_PARAM_VALUE "INFO3"

Definition at line 129 of file logger.hpp.

9.28.1.12 #define INFO_PARAM_VALUE "INFO"

Definition at line 126 of file logger.hpp.

9.28.1.13 #define LINE_STRING STRINGIZE(__LINE__)

Definition at line 78 of file logger.hpp.

9.28.1.14 #define LOG_DEBUG LOGGER_DEBUG(debug_levels_enum::DEBUG)

Definition at line 111 of file logger.hpp.

9.28.1.15 #define LOG_DEBUG1 LOGGER_DEBUG(debug_levels_enum::DEBUG1)

Definition at line 112 of file logger.hpp.

9.28.1.16 #define LOG_DEBUG2 LOGGER_DEBUG(debug_levels_enum::DEBUG2)

Definition at line 113 of file logger.hpp.

9.28.1.17 #define LOG_DEBUG3 LOGGER_DEBUG(debug_levels_enum::DEBUG3)

Definition at line 114 of file logger.hpp.

9.28.1.18 #define LOG_DEBUG4 LOGGER_DEBUG(debug_levels_enum::DEBUG4)

Definition at line 115 of file logger.hpp.

9.28.1.19 #define LOG_ERROR LOGGER(debug_levels_enum::ERROR)

Definition at line 103 of file logger.hpp.

9.28.1.20 #define LOG_INFO LOGGER(debug_levels_enum::INFO)

Definition at line 107 of file logger.hpp.

9.28.1.21 #define LOG_INFO1 LOGGER(debug_levels_enum::INFO1)

Definition at line 108 of file logger.hpp.

9.28.1.22 #define LOG_INFO2 LOGGER(debug_levels_enum::INFO2)

Definition at line 109 of file logger.hpp.

9.28.1.23 #define LOG_INFO3 LOGGER(debug_levels_enum::INFO3)

Definition at line 110 of file logger.hpp.

9.28.1.24 #define LOG_RESULT LOGGER(debug_levels_enum::RESULT)

Definition at line 106 of file logger.hpp.

9.28.1.25 #define LOG_USAGE LOGGER(debug_levels_enum::USAGE)

Definition at line 105 of file logger.hpp.

9.28.1.26 #define LOG_WARNING LOGGER(debug_levels_enum::WARNING)

Definition at line 104 of file logger.hpp.

9.28.1.27 #define LOGGER(*level*)

Value:

```
{                                                \
  logging_synch::rec_scoped_lock lock(logging_synch::mv);\
  if (level > LOGER_M_GRAM_LEVEL_MAX) ;                  \
  else if (level > logger::get_reporting_level()) ;  \
       else logger::get(level)
```

Definition at line 83 of file logger.hpp.

9.28.1.28 #define LOGGER_DEBUG(*level*)

Value:

```
{                                                \
  logging_synch::rec_scoped_lock lock(logging_synch::mv);\
  if (level > LOGER_M_GRAM_LEVEL_MAX) ;                  \
  else if (level > logger::get_reporting_level()) ;  \
       else logger::get(level, __FILENAME__, __FUNCTION__,
      LINE_STRING)
```

Definition at line 92 of file logger.hpp.

9.28.1.29 #define PROGRESS_UPDATE_PERIOD 0.05

Definition at line 81 of file logger.hpp.

9.28.1.30 #define RESULT_PARAM_VALUE "RESULT"

Definition at line 125 of file logger.hpp.

9.28.1.31 #define SSTR(*x*) std::dec << (x)

Definition at line 73 of file logger.hpp.

9.28.1.32 #define STRINGIZE(*x*) STRINGIZE2(x)

Definition at line 76 of file logger.hpp.

9.28.1.33 #define STRINGIZE2(*x*) #x

Definition at line 77 of file logger.hpp.

9.28.1.34 #define USAGE_PARAM_VALUE "USAGE"

Definition at line 124 of file logger.hpp.

9.28.1.35 #define WARNING_PARAM_VALUE "WARN"

Definition at line 123 of file logger.hpp.

9.28.1.36 #define WHITE_SPACE_SEPARATOR " "

Definition at line 137 of file logger.hpp.

9.29 inc/common/utils/math_utils.hpp File Reference

```
#include <cstdint>
#include <cstring>
#include <bitset>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
```

Namespaces

- uva
- uva::utils
- uva::utils::math
- uva::utils::math::const_expr
- uva::utils::math::log2
- uva::utils::math::bits

Macros

- #define BYTE_IDX(total_bit_pos) ((total_bit_pos) / NUM_BITS_IN_UINT_8)
- #define REMAINING_BIT_IDX(total_bit_pos) ((total_bit_pos) % NUM_BITS_IN_UINT_8)
- #define NUM_FULL_BYTES(number_of_bits) ((number_of_bits) / NUM_BITS_IN_UINT_8)
- #define NUM_BITS_REMAINDER(number_of_bits) ((number_of_bits) % NUM_BITS_IN_UINT_8)
- #define NUM_BYTES_4_BITS(number_of_bits) (((number_of_bits) + (NUM_BITS_IN_UINT_8 - 1)) / NUM↩ _BITS_IN_UINT_8)
- #define BYTES_TO_BITS(number_of_bytes) ((number_of_bytes) ∗ NUM_BITS_IN_UINT_8)
- #define VALUE_LEN_BYTES(VALUE) static_cast<uint8_t> (const_expr::ceil(const_expr::log2(VALUE)/8))
- #define HANDLE_ENDIAN(value_type, value)

Functions

- constexpr double uva::utils::math::const_expr::log2 (double value, double pow=0.0)
- constexpr uint64_t uva::utils::math::const_expr::ceil (double value)
- constexpr uint64_t uva::utils::math::const_expr::power (uint64_t value, uint8_t pow)

9.29.1 Macro Definition Documentation

9.29.1.1 #define BYTE_IDX(*total_bit_pos*) ((total_bit_pos) / NUM_BITS_IN_UINT_8)

Definition at line 136 of file math_utils.hpp.

9.29.1.2 #define BYTES_TO_BITS(*number_of_bytes*) ((number_of_bytes) * NUM_BITS_IN_UINT_8)

Definition at line 147 of file math_utils.hpp.

9.29.1.3 #define HANDLE_ENDIAN(*value_type, value*)

Value:

```
if(sizeof(value_type) == 2) { \
    (value) = __builtin_bswap16((value)); \
} else { \
    if(sizeof(value_type) == 4) { \
        (value) = __builtin_bswap32((value)); \
    } else { \
        if(sizeof(value_type) == 8) { \
            (value) = __builtin_bswap64((value)); \
        } else { \
            throw uva_exception("HANDLE_ENDIAN(value_type, value): Unsupported value type!");
      \
        } \
    } \
}
```

Definition at line 156 of file math_utils.hpp.

9.29.1.4 #define NUM_BITS_REMAINDER(*number_of_bits*) ((number_of_bits) % NUM_BITS_IN_UINT_8)

Definition at line 143 of file math_utils.hpp.

9.29.1.5 #define NUM_BYTES_4_BITS(*number_of_bits*) (((number_of_bits) + (NUM_BITS_IN_UINT_8 - 1)) / NUM_BITS_IN_UINT_8)

Definition at line 145 of file math_utils.hpp.

9.29.1.6 #define NUM_FULL_BYTES(*number_of_bits*) ((number_of_bits) / NUM_BITS_IN_UINT_8)

Definition at line 141 of file math_utils.hpp.

9.29.1.7 #define REMAINING_BIT_IDX(*total_bit_pos*) ((total_bit_pos) % NUM_BITS_IN_UINT_8)

Definition at line 138 of file math_utils.hpp.

9.29.1.8 #define VALUE_LEN_BYTES(*VALUE*) static_cast<uint8_t> (const_expr::ceil(const_expr::log2(VALUE)/8))

Definition at line 149 of file math_utils.hpp.

9.30 inc/common/utils/monitor/statistics_monitor.hpp File Reference

```
#include "common/utils/exceptions.hpp"
```

Classes

- struct uva::utils::monitor::memory_usage
- class uva::utils::monitor::stat_monitor

Namespaces

- uva
- uva::utils
- uva::utils::monitor

Typedefs

- typedef memory_usage uva::utils::monitor::TMemotyUsage

Variables

- const uint32_t uva::utils::monitor::BYTES_ONE_MB = 1024u

9.31 inc/common/utils/string_utils.hpp File Reference

```
#include <locale>
#include <string>
#include <vector>
#include <sstream>
#include <cstddef>
#include <limits>
#include <cstring>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
```

Namespaces

- uva
- uva::utils
- uva::utils::text

Macros

- #define valid_digit(c) ((c) >= '0' && (c) <= '9')

Variables

- const string uva::utils::text::UTF8_ASCII_WHITESPACES = u8"\t\f\v\n\r "
- const string uva::utils::text::UTF8_ASCII_PUNCTUATIONS = u8".,?!/'\"‘@#$%^&∗()[ ]{}-_+=∗<>∼|\\;:"
- const char uva::utils::text::ASCII_SPACE_CHAR = ' '
- const string uva::utils::text::UTF8_SPACE_STRING = u8" "
- const string uva::utils::text::UTF8_EMPTY_STRING = u8""
- const string uva::utils::text::UTF8_NEW_LINE_STRING = u8"\n"

9.31.1 Macro Definition Documentation

9.31.1.1 #define valid_digit(c) ((c) >= '0' && (c) <= '9')

Definition at line 318 of file string_utils.hpp.

9.32 inc/common/utils/threads.hpp File Reference

```
#include <atomic>
#include <thread>
#include <mutex>
#include <condition_variable>
#include <functional>
```

Namespaces

- uva
- uva::utils
- uva::utils::threads

Typedefs

- typedef lock_guard< recursive_mutex > uva::utils::threads::recursive_guard
- typedef lock_guard< mutex > uva::utils::threads::scoped_guard
- typedef unique_lock< mutex > uva::utils::threads::unique_guard
- typedef atomic< bool > uva::utils::threads::a_bool_flag
- typedef const a_bool_flag & uva::utils::threads::acr_bool_flag

9.33 inc/main.hpp File Reference

```
#include <string>
#include <stdexcept>
#include <execinfo.h>
#include <INI.h>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::common

Macros

- #define SAFE_DESTROY(ptr)
- #define MAX_STACK_TRACE_LEN 100
- #define GET_ASSERT(ini, section, key, value_str)

Functions

- template<typename INT_TYPE >
  INT_TYPE uva::smt::bpbd::common::get_integer (INI<> &ini, string section, string key)
- string uva::smt::bpbd::common::get_string (INI<> &ini, string section, string key)
- float uva::smt::bpbd::common::get_float (INI<> &ini, string section, string key)

9.33.1 Macro Definition Documentation

9.33.1.1 #define GET_ASSERT(*ini, section, key, value_str*)

Value:

```
const string value_str = ini.get(section, key, UNKNOWN_INI_FILE_VALUE); \
    ASSERT_CONDITION_THROW((value_str == UNKNOWN_INI_FILE_VALUE), \
            string("Could not find '[") + section + string("]/") + \
            key + string("' section/key in the configuration file!"));
```

Definition at line 98 of file main.hpp.

9.33.1.2 #define MAX_STACK_TRACE_LEN 100

Definition at line 77 of file main.hpp.

9.33.1.3 #define SAFE_DESTROY(*ptr*)

Value:

```
if (ptr != NULL) { \
        delete ptr; \
        ptr = NULL; \
    }
```

Definition at line 47 of file main.hpp.

9.34 inc/server/cmd_line_handler.hpp File Reference

```
#include "common/utils/logging/logger.hpp"
#include "server/server_parameters.hpp"
#include "server/translation_server.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server

Functions

- void uva::smt::bpbd::server::stop (translation_server &server, thread &server_thread)
- void uva::smt::bpbd::server::print_the_prompt ()
- void uva::smt::bpbd::server::print_server_commands ()

- bool uva::smt::bpbd::server::begins_with (const string &str, const string &prefix)
- string uva::smt::bpbd::server::get_string_value (const string &str, const string &prefix)
- int32_t uva::smt::bpbd::server::get_int_value (const string &str, const string &prefix)
- float uva::smt::bpbd::server::get_float_value (const string &str, const string &prefix)
- void uva::smt::bpbd::server::set_log_level (const string &cmd, const string &prefix)
- void uva::smt::bpbd::server::set_num_threads (server_parameters &params, translation_server &server, const string &cmd, const string &prefix)
- void uva::smt::bpbd::server::set_decoder_params (const string &cmd, de_parameters &de_params)
- bool uva::smt::bpbd::server::process_input_cmd (server_parameters &params, translation_server &server, thread &server_thread, char command[CMD_BUFF_SIZE])
- void uva::smt::bpbd::server::perform_command_loop (server_parameters &params, translation_server &server, thread &server_thread)

9.35 inc/server/common/models/phrase_uid.hpp File Reference

```
#include <string>
#include <vector>
#include <cstdint>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/hashing_utils.hpp"
#include "common/utils/string_utils.hpp"
#include "server/server_configs.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::common
- uva::smt::bpbd::server::common::models

9.36 inc/server/decoder/de_configs.hpp File Reference

```
#include <inttypes.h>
#include <cstdint>
#include <string>
#include "server/server_configs.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::decoder

9.37 inc/server/decoder/de_configurator.hpp File Reference

```
#include "server/decoder/de_parameters.hpp"
#include "server/decoder/sentence/sentence_decoder.hpp"
```

Classes

- class uva::smt::bpbd::server::decoder::de_configurator

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::decoder

9.38 inc/server/decoder/de_parameters.hpp File Reference

```
#include <string>
#include <ostream>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/threads.hpp"
#include "server/decoder/de_configs.hpp"
```

Classes

- struct uva::smt::bpbd::server::decoder::de_parameters_struct

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::decoder

Typedefs

- typedef de_parameters_struct uva::smt::bpbd::server::decoder::de_parameters

9.39 inc/server/decoder/sentence/sentence_data_map.hpp File Reference

```
#include <string>
```

```
#include <cstdint>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/containers/upp_diag_matrix.hpp"
#include "server/common/models/phrase_uid.hpp"
#include "server/decoder/de_configs.hpp"
#include "server/tm/models/tm_source_entry.hpp"
```

Classes

- struct uva::smt::bpbd::server::decoder::sentence::phrase_data_entry

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::decoder
- uva::smt::bpbd::server::decoder::sentence

Typedefs

- typedef upp_diag_matrix< phrase_data_entry > uva::smt::bpbd::server::decoder::sentence::sentence_data_map

9.40 inc/server/decoder/sentence/sentence_decoder.hpp File Reference

```
#include <algorithm>
#include "common/utils/threads.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/string_utils.hpp"
#include "server/common/models/phrase_uid.hpp"
#include "server/decoder/de_parameters.hpp"
#include "server/decoder/sentence/sentence_data_map.hpp"
#include "server/decoder/stack/multi_stack.hpp"
#include "server/lm/lm_configurator.hpp"
#include "server/tm/tm_configurator.hpp"
#include "server/rm/rm_configurator.hpp"
```

Classes

- class uva::smt::bpbd::server::decoder::sentence::sentence_decoder

Namespaces

- uva
- uva::smt
- uva::smt::bpbd

- uva::smt::bpbd::server
- uva::smt::bpbd::server::decoder
- uva::smt::bpbd::server::decoder::sentence

9.41 inc/server/decoder/stack/multi_stack.hpp File Reference

```
#include <string>
#include <functional>
#include "common/utils/threads.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "server/lm/lm_configurator.hpp"
#include "server/rm/proxy/rm_query_proxy.hpp"
#include "server/decoder/de_configs.hpp"
#include "server/decoder/de_parameters.hpp"
#include "server/decoder/stack/stack_level.hpp"
#include "server/decoder/stack/stack_data.hpp"
```

Classes

- class uva::smt::bpbd::server::decoder::stack::multi_stack

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::decoder
- uva::smt::bpbd::server::decoder::stack

9.42 inc/server/decoder/stack/stack_data.hpp File Reference

```
#include <functional>
#include "common/utils/threads.hpp"
#include "server/decoder/sentence/sentence_data_map.hpp"
#include "server/rm/proxy/rm_query_proxy.hpp"
#include "server/lm/proxy/lm_fast_query_proxy.hpp"
```

Classes

- class uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MA↩ X_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >
- struct uva::smt::bpbd::server::decoder::stack::stack_data

Namespaces

- uva
- uva::smt

- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::decoder
- uva::smt::bpbd::server::decoder::stack

Typedefs

- typedef stack_state_templ< MAX_WORDS_PER_SENTENCE, LM_HISTORY_LEN_MAX, LM_MAX_QUERY_LEN > uva::smt::bpbd::server::decoder::stack::stack_state
- typedef stack_state * uva::smt::bpbd::server::decoder::stack::stack_state_ptr
- typedef function< void(stack_state_ptr) > uva::smt::bpbd::server::decoder::stack::add_new_state_function

9.43 inc/server/decoder/stack/stack_level.hpp File Reference

```
#include <string>
#include "common/utils/threads.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "server/decoder/de_configs.hpp"
#include "server/decoder/de_parameters.hpp"
#include "server/decoder/stack/stack_state.hpp"
```

Classes

- class uva::smt::bpbd::server::decoder::stack::stack_level

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::decoder
- uva::smt::bpbd::server::decoder::stack

Typedefs

- typedef stack_level * uva::smt::bpbd::server::decoder::stack::stack_level_ptr

9.44 inc/server/decoder/stack/stack_state.hpp File Reference

```
#include <vector>
#include <algorithm>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/string_utils.hpp"
#include "server/lm/lm_configurator.hpp"
#include "server/rm/proxy/rm_query_proxy.hpp"
#include "server/decoder/de_configs.hpp"
#include "server/decoder/de_parameters.hpp"
#include "server/decoder/stack/state_data.hpp"
```

Classes

- class uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::decoder
- uva::smt::bpbd::server::decoder::stack

9.45 inc/server/decoder/stack/state_data.hpp File Reference

```
#include <string>
#include <bitset>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/containers/circular_queue.hpp"
#include "server/tm/models/tm_target_entry.hpp"
#include "server/rm/models/rm_entry.hpp"
#include "server/decoder/stack/stack_data.hpp"
```

Classes

- struct uva::smt::bpbd::server::decoder::stack::state_data_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::decoder
- uva::smt::bpbd::server::decoder::stack

9.46 inc/server/lm/builders/lm_basic_builder.hpp File Reference

```
#include <regex>
#include "server/lm/lm_consts.hpp"
#include "server/lm/lm_parameters.hpp"
#include "common/utils/file/text_piece_reader.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::arpa::lm_basic_builder< trie_type, reader_type >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::arpa

9.47 inc/server/lm/builders/lm_gram_builder.hpp File Reference

```
#include <regex>
#include <functional>
#include "server/lm/lm_consts.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/file/memory_mapped_file_reader.hpp"
#include "server/lm/lm_parameters.hpp"
#include "server/lm/mgrams/model_m_gram.hpp"
```

Classes

- struct uva::smt::bpbd::server::lm::arpa::TAddGramFunct< WordIndexType >
- class uva::smt::bpbd::server::lm::arpa::lm_gram_builder< WordIndexType, CURR_LEVEL, is_mult_weight >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::arpa

9.48 inc/server/lm/builders/lm_gram_builder_factory.hpp File Reference

```
#include <string>
#include <ios>
#include <functional>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "server/lm/lm_consts.hpp"
#include "server/lm/lm_configs.hpp"
#include "server/lm/lm_parameters.hpp"
#include "server/lm/builders/lm_gram_builder.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::arpa::lm_gram_builder_factory< TrieType >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::arpa

9.49 inc/server/lm/dictionaries/aword_index.hpp File Reference

```
#include <string>
#include "server/lm/lm_consts.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/file/text_piece_reader.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::dictionary::aword_index

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::dictionary

9.50 inc/server/lm/dictionaries/basic_word_index.hpp File Reference

```
#include <string>
#include <unordered_map>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/hashing_utils.hpp"
#include "common/utils/file/text_piece_reader.hpp"
#include "common/utils/containers/greedy_memory_allocator.hpp"
#include "server/lm/lm_consts.hpp"
#include "server/lm/dictionaries/aword_index.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::dictionary::basic_word_index

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::dictionary

9.51 inc/server/lm/dictionaries/counting_word_index.hpp File Reference

```
#include <string>
#include "basic_word_index.hpp"
#include "server/lm/lm_consts.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/containers/array_utils.hpp"
#include "common/utils/file/text_piece_reader.hpp"
```

Classes

- struct uva::smt::bpbd::server::lm::dictionary::__counting_word_index::TWordInfo
- class uva::smt::bpbd::server::lm::dictionary::counting_word_index

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::dictionary
- uva::smt::bpbd::server::lm::dictionary::__counting_word_index

Functions

- bool uva::smt::bpbd::server::lm::dictionary::__counting_word_index::operator< (const TWordInfo &one, const TWordInfo &two)

9.52 inc/server/lm/dictionaries/hashing_word_index.hpp File Reference

```
#include <string>
#include "server/lm/lm_consts.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "aword_index.hpp"
#include "common/utils/hashing_utils.hpp"
#include "common/utils/file/text_piece_reader.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::dictionary::hashing_word_index

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::dictionary

9.53 inc/server/lm/dictionaries/optimizing_word_index.hpp File Reference

```
#include <string>
#include <cstring>
#include "server/lm/lm_consts.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/math_utils.hpp"
#include "server/lm/dictionaries/aword_index.hpp"
#include "server/lm/dictionaries/basic_word_index.hpp"
#include "common/utils/containers/array_utils.hpp"
#include "common/utils/file/text_piece_reader.hpp"
```

Classes

- struct uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index::word_index_bucket_entry< word_id_type >
- class uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::dictionary
- uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index

Macros

- #define IS_EQUAL(token, entry) (((token).length() == (entry).m_len) && (strncmp((token).get_begin_c_str(), (entry).m_word, (entry).m_len) == 0))

Typedefs

- typedef optimizing_word_index< basic_word_index > uva::smt::bpbd::server::lm::dictionary::basic_optimizing_word_index
- typedef optimizing_word_index< counting_word_index > uva::smt::bpbd::server::lm::dictionary::counting_optimizing_word_index

Functions

- template<typename word_id_type >
  struct uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index::word_index_bucket_entry uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index::__attribute__ ((packed))
- word_index_bucket_entry ()

Variables

- char * m_word
- uint8_t m_len
- word_id_type m_word_id

9.53.1 Macro Definition Documentation

9.53.1.1 #define IS_EQUAL(*token, entry*) (((token).length() == (entry).m_len) && (strncmp((token).get_begin_c_str(), (entry).m_word, (entry).m_len) == 0))

Definition at line 140 of file optimizing_word_index.hpp.

9.53.2 Function Documentation

9.53.2.1 __attribute__::word_index_bucket_entry ( )

Definition at line 417 of file optimizing_word_index.hpp.

9.53.3 Variable Documentation

9.53.3.1 uint8_t m_len

Definition at line 420 of file optimizing_word_index.hpp.

9.53.3.2 char* m_word

Definition at line 419 of file optimizing_word_index.hpp.

9.53.3.3 word_id_type m_word_id

Definition at line 421 of file optimizing_word_index.hpp.

9.54 inc/server/lm/lm_configs.hpp File Reference

```
#include <inttypes.h>
#include <string>
#include "server/server_configs.hpp"
#include "common/utils/file/cstyle_file_reader.hpp"
#include "common/utils/file/file_stream_reader.hpp"
#include "common/utils/file/memory_mapped_file_reader.hpp"
#include "server/lm/dictionaries/basic_word_index.hpp"
#include "server/lm/dictionaries/counting_word_index.hpp"
#include "server/lm/dictionaries/optimizing_word_index.hpp"
#include "server/lm/dictionaries/hashing_word_index.hpp"
#include "server/lm/builders/lm_basic_builder.hpp"
#include "server/lm/models/c2d_hybrid_trie.hpp"
#include "server/lm/models/c2d_map_trie.hpp"
#include "server/lm/models/c2w_array_trie.hpp"
#include "server/lm/models/g2d_map_trie.hpp"
#include "server/lm/models/h2d_map_trie.hpp"
#include "server/lm/models/w2c_array_trie.hpp"
#include "server/lm/models/w2c_hybrid_trie.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

Typedefs

- typedef hashing_word_index uva::smt::bpbd::server::lm::lm_word_index
- typedef h2d_map_trie< lm_word_index > uva::smt::bpbd::server::lm::lm_model_type
- typedef cstyle_file_reader uva::smt::bpbd::server::lm::lm_model_reader
- typedef lm_basic_builder< lm_model_type, lm_model_reader > uva::smt::bpbd::server::lm::lm_builder_type

9.55 inc/server/lm/lm_configurator.hpp File Reference

```
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "server/lm/lm_parameters.hpp"
#include "server/lm/proxy/lm_proxy.hpp"
#include "server/lm/proxy/lm_proxy_local.hpp"
#include "server/lm/proxy/lm_slow_query_proxy.hpp"
#include "server/lm/proxy/lm_fast_query_proxy.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::lm_configurator

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

Variables

- class uva::smt::bpbd::server::lm::lm_configurator uva::smt::bpbd::server::lm::__attribute__

9.56 inc/server/lm/lm_consts.hpp File Reference

```
#include <inttypes.h>
#include <string>
#include "server/server_configs.hpp"
#include "server/server_consts.hpp"
#include "common/utils/containers/dynamic_memory_arrays.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::dictionary
- uva::smt::bpbd::server::lm::dictionary::__AWordIndex
- uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index
- uva::smt::bpbd::server::lm::identifiers
- uva::smt::bpbd::server::lm::__C2DHybridTrie
- uva::smt::bpbd::server::lm::__C2DMapTrie
- uva::smt::bpbd::server::lm::__G2DMapTrie
- uva::smt::bpbd::server::lm::__H2DMapTrie
- uva::smt::bpbd::server::lm::__W2CArrayTrie
- uva::smt::bpbd::server::lm::__C2WArrayTrie
- uva::smt::bpbd::server::lm::__W2CHybridTrie
- uva::smt::bpbd::server::lm::m_grams

Typedefs

- typedef uint32_t uva::smt::bpbd::server::lm::identifiers::TShortId
- typedef uint64_t uva::smt::bpbd::server::lm::identifiers::TLongId

9.57 inc/server/lm/lm_executor.hpp File Reference

```
#include <string>
#include <vector>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "server/lm/lm_consts.hpp"
#include "server/lm/lm_parameters.hpp"
#include "server/lm/lm_configurator.hpp"
#include "server/lm/proxy/lm_fast_query_proxy.hpp"
#include "server/lm/dictionaries/basic_word_index.hpp"
#include "server/lm/dictionaries/counting_word_index.hpp"
#include "server/lm/dictionaries/optimizing_word_index.hpp"
#include "server/lm/dictionaries/hashing_word_index.hpp"
#include "server/lm/builders/lm_basic_builder.hpp"
#include "server/lm/builders/lm_gram_builder.hpp"
#include "server/lm/models/m_gram_query.hpp"
```

Classes

- struct uva::smt::bpbd::server::lm::__executor::lm_exec_params

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::__executor

9.58 inc/server/lm/lm_parameters.hpp File Reference

```
#include <string>
#include <ostream>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/string_utils.hpp"
#include "server/server_configs.hpp"
```

Classes

- struct uva::smt::bpbd::server::lm::lm_parameters

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

9.59 inc/server/lm/mgrams/m_gram_id.hpp File Reference

```
#include <string>
#include "server/lm/lm_consts.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/file/text_piece_reader.hpp"
#include "common/utils/hashing_utils.hpp"
#include "common/utils/math_utils.hpp"
#include "m_gram_id_tables.hpp"
```

Classes

- struct uva::smt::bpbd::server::lm::m_grams::m_gram_id::T_Gram_Id_Key
- class uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::m_grams
- uva::smt::bpbd::server::lm::m_grams::m_gram_id

Macros

- #define N_GRAM_ID_TYPE_LEN_BYTES(LEVEL) VALUE_LEN_BYTES(NUMBER_ID_TYPES_PER_L↩ EVEL[(LEVEL)])
- #define MAX_N_GRAM_ID_LEN_BYTES(LEVEL) static_cast<uint8_t> ((LEVEL) * NUM_BYTES_WOR↩ D_ID + ID_TYPE_LEN_BYTES[LEVEL])
- #define DECLARE_STACK_GRAM_ID(type, name, level) uint8_t name[type::MAX_ID_LEN_BYTES[(level)]];

Typedefs

- typedef uint8_t * uva::smt::bpbd::server::lm::m_grams::m_gram_id::TM_Gram_Id_Value_Ptr

9.59.1 Macro Definition Documentation

9.59.1.1 #define DECLARE_STACK_GRAM_ID(*type, name, level*) uint8_t name[type::MAX_ID_LEN_BYTES[(level)]];

Definition at line 158 of file m_gram_id.hpp.

9.59.1.2 #define MAX_N_GRAM_ID_LEN_BYTES(*LEVEL*) static_cast<uint8_t> ((LEVEL) * NUM_BYTES_WORD_ID + ID_TYPE_LEN_BYTES[LEVEL])

Definition at line 140 of file m_gram_id.hpp.

9.59.1.3 #define N_GRAM_ID_TYPE_LEN_BYTES(*LEVEL*) VALUE_LEN_BYTES(NUMBER_ID_TYPES_PER_LEVEL[(LE↩ VEL)])

Definition at line 122 of file m_gram_id.hpp.

9.60 inc/server/lm/mgrams/m_gram_id_tables.hpp File Reference

Macros

- #define BYTE_M_GRAM_ID_TABLES_HPP

9.60.1 Macro Definition Documentation

9.60.1.1 #define BYTE_M_GRAM_ID_TABLES_HPP

Definition at line 28 of file m_gram_id.hpp.

9.61 inc/server/lm/mgrams/m_gram_payload.hpp File Reference

```
#include <string>
#include <ostream>
#include "server/lm/lm_consts.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/file/text_piece_reader.hpp"
#include "common/utils/hashing_utils.hpp"
#include "common/utils/math_utils.hpp"
#include "server/lm/mgrams/m_gram_id.hpp"
#include "server/lm/dictionaries/basic_word_index.hpp"
#include "server/lm/dictionaries/counting_word_index.hpp"
#include "server/lm/dictionaries/optimizing_word_index.hpp"
```

Classes

- struct uva::smt::bpbd::server::lm::m_grams::m_gram_payload_s
- class uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID↩ _LENGTH >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::m_grams

Typedefs

- typedef m_gram_payload_s uva::smt::bpbd::server::lm::m_grams::m_gram_payload

9.62 inc/server/lm/mgrams/model_m_gram.hpp File Reference

```
#include <string>
#include "server/lm/lm_consts.hpp"
#include "common/utils/exceptions.hpp"
#include "server/lm/mgrams/m_gram_payload.hpp"
#include "common/utils/file/text_piece_reader.hpp"
#include "common/utils/hashing_utils.hpp"
#include "common/utils/math_utils.hpp"
#include "server/common/models/phrase_uid.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::m_grams::model_m_gram

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::m_grams

9.63 inc/server/lm/mgrams/query_m_gram.hpp File Reference

```
#include <string>
#include "server/lm/lm_consts.hpp"
#include "common/utils/file/text_piece_reader.hpp"
#include "common/utils/hashing_utils.hpp"
#include "common/utils/math_utils.hpp"
#include "common/utils/exceptions.hpp"
#include "server/common/models/phrase_uid.hpp"
#include "m_gram_payload.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::m_grams::query_m_gram

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::m_grams

9.64 inc/server/lm/models/bitmap_hash_cache.hpp File Reference

```
#include <cstdint>
#include <bitset>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "server/lm/mgrams/model_m_gram.hpp"
#include "common/utils/hashing_utils.hpp"
#include "common/utils/math_utils.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::caching::BitmapHashCache

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::caching

9.65 inc/server/lm/models/c2d_hybrid_trie.hpp File Reference

```
#include <utility>
#include <unordered_map>
#include "server/lm/lm_consts.hpp"
#include "common/utils/logging/logger.hpp"
#include "layered_trie_base.hpp"
#include "server/lm/dictionaries/hashing_word_index.hpp"
#include "common/utils/containers/greedy_memory_allocator.hpp"
#include "common/utils/hashing_utils.hpp"
#include "common/utils/file/text_piece_reader.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::c2d_hybrid_trie< WordIndexType >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

Typedefs

- typedef c2d_hybrid_trie< basic_word_index > uva::smt::bpbd::server::lm::TC2DHybridTrieBasic
- typedef c2d_hybrid_trie< counting_word_index > uva::smt::bpbd::server::lm::TC2DHybridTrieCount
- typedef c2d_hybrid_trie< basic_optimizing_word_index > uva::smt::bpbd::server::lm::TC2DHybridTrieOpt↩ Basic
- typedef c2d_hybrid_trie< counting_optimizing_word_index > uva::smt::bpbd::server::lm::TC2DHybridTrie↩ OptCount
- typedef c2d_hybrid_trie< hashing_word_index > uva::smt::bpbd::server::lm::TC2DHybridTrieHashing

9.66 inc/server/lm/models/c2d_map_trie.hpp File Reference

```
#include <utility>
#include <unordered_map>
#include "server/lm/lm_consts.hpp"
#include "common/utils/logging/logger.hpp"
#include "layered_trie_base.hpp"
#include "common/utils/containers/greedy_memory_allocator.hpp"
#include "common/utils/hashing_utils.hpp"
#include "common/utils/file/text_piece_reader.hpp"
#include "server/lm/dictionaries/hashing_word_index.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::c2d_map_trie< WordIndexType >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

Typedefs

- typedef c2d_map_trie< basic_word_index > uva::smt::bpbd::server::lm::TC2DMapTrieBasic
- typedef c2d_map_trie< counting_word_index > uva::smt::bpbd::server::lm::TC2DMapTrieCount
- typedef c2d_map_trie< hashing_word_index > uva::smt::bpbd::server::lm::TC2DMapTrieHashing
- typedef c2d_map_trie< basic_optimizing_word_index > uva::smt::bpbd::server::lm::TC2DMapTrieOptBasic
- typedef c2d_map_trie< counting_optimizing_word_index > uva::smt::bpbd::server::lm::TC2DMapTrieOpt↩ Count

9.67 inc/server/lm/models/c2w_array_trie.hpp File Reference

```
#include <string>
#include "server/lm/lm_consts.hpp"
#include "common/utils/logging/logger.hpp"
#include "layered_trie_base.hpp"
#include "server/lm/dictionaries/aword_index.hpp"
#include "server/lm/dictionaries/hashing_word_index.hpp"
#include "common/utils/containers/array_utils.hpp"
```

Classes

- struct uva::smt::bpbd::server::lm::__C2WArrayTrie::TWordIdPBData
- struct uva::smt::bpbd::server::lm::__C2WArrayTrie::TCtxIdProbData
- class uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >
- struct uva::smt::bpbd::server::lm::c2w_array_trie< WordIndexType >::TSubArrReference

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::__C2WArrayTrie

Typedefs

- typedef c2w_array_trie< basic_word_index > uva::smt::bpbd::server::lm::TC2WArrayTrieBasic
- typedef c2w_array_trie< counting_word_index > uva::smt::bpbd::server::lm::TC2WArrayTrieCount
- typedef c2w_array_trie< basic_optimizing_word_index > uva::smt::bpbd::server::lm::TC2WArrayTrieOpt↩Basic
- typedef c2w_array_trie< counting_optimizing_word_index > uva::smt::bpbd::server::lm::TC2WArrayTrie↩OptCount
- typedef c2w_array_trie< hashing_word_index > uva::smt::bpbd::server::lm::TC2WArrayTrieHashing

Functions

- bool uva::smt::bpbd::server::lm::__C2WArrayTrie::operator< (const TWordIdPBData &one, const TWordId↩PBData &two)
- int8_t uva::smt::bpbd::server::lm::__C2WArrayTrie::compare (const TCtxIdProbData &one, const TCtxId↩ProbData &two)
- bool uva::smt::bpbd::server::lm::__C2WArrayTrie::operator< (const TCtxIdProbData &one, const TCtxId↩ProbData &two)
- bool uva::smt::bpbd::server::lm::__C2WArrayTrie::operator> (const TCtxIdProbData &one, const TCtxId↩ProbData &two)
- bool uva::smt::bpbd::server::lm::__C2WArrayTrie::operator== (const TCtxIdProbData &one, const TCtxId↩ProbData &two)

9.68 inc/server/lm/models/g2d_map_trie.hpp File Reference

```
#include <string>
```

```
#include "server/lm/lm_consts.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "server/lm/dictionaries/aword_index.hpp"
#include "server/lm/dictionaries/hashing_word_index.hpp"
#include "server/lm/mgrams/model_m_gram.hpp"
#include "server/lm/mgrams/m_gram_id.hpp"
#include "common/utils/file/text_piece_reader.hpp"
#include "common/utils/containers/array_utils.hpp"
#include "common/utils/containers/fixed_size_hashmap.hpp"
#include "generic_trie_base.hpp"
#include "w2c_array_trie.hpp"
```

Classes

- struct uva::smt::bpbd::server::lm::__G2DMapTrie::S_M_GramData< TPayloadType, TWordIdType >
- class uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::__G2DMapTrie

Typedefs

- typedef g2d_map_trie< basic_word_index > uva::smt::bpbd::server::lm::TG2DMapTrieBasic
- typedef g2d_map_trie< counting_word_index > uva::smt::bpbd::server::lm::TG2DMapTrieCount
- typedef g2d_map_trie< basic_optimizing_word_index > uva::smt::bpbd::server::lm::TG2DMapTrieOptBasic
- typedef g2d_map_trie< counting_optimizing_word_index > uva::smt::bpbd::server::lm::TG2DMapTrieOpt↩ Count
- typedef g2d_map_trie< hashing_word_index > uva::smt::bpbd::server::lm::TG2DMapTrieHashing

9.69 inc/server/lm/models/generic_trie_base.hpp File Reference

```
#include <string>
#include "server/lm/lm_consts.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "server/lm/mgrams/model_m_gram.hpp"
#include "server/lm/mgrams/query_m_gram.hpp"
#include "common/utils/file/text_piece_reader.hpp"
#include "server/lm/dictionaries/basic_word_index.hpp"
#include "server/lm/dictionaries/counting_word_index.hpp"
#include "server/lm/dictionaries/optimizing_word_index.hpp"
#include "server/lm/dictionaries/hashing_word_index.hpp"
#include "server/lm/models/m_gram_query.hpp"
#include "server/lm/models/word_index_trie_base.hpp"
#include "server/lm/models/bitmap_hash_cache.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE↩ _BUCKETS_FACTOR >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

Macros

- #define REPORT_COLLISION_WARNING(gram, word_id, contextId, prevProb, prevBackOff, newProb, newBackOff)
- #define INSTANTIATE_TRIE_FUNCS_LEVEL(LEVEL, TRIE_TYPE_NAME, ...) template void TRIE_TYP↩ E_NAME<__VA_ARGS__>::add_m_gram<LEVEL>(const model_m_gram & gram);
- #define INSTANTIATE_TRIE_TEMPLATE_TYPE(TRIE_TYPE_NAME, ...)

Enumerations

- enum uva::smt::bpbd::server::lm::MGramStatusEnum { uva::smt::bpbd::server::lm::UNDEFINED_MGS = 0, uva::smt::bpbd::server::lm::BAD_END_WORD_UNKNOWN_MGS = 1, uva::smt::bpbd::server::lm::BAD_N↩ O_PAYLOAD_MGS = 2, uva::smt::bpbd::server::lm::GOOD_PRESENT_MGS = 3 }

9.69.1 Macro Definition Documentation

9.69.1.1 #define INSTANTIATE_TRIE_FUNCS_LEVEL(*LEVEL, TRIE_TYPE_NAME, ...*) template void TRIE_TYPE_NAME<__VA_ARGS__>::add_m_gram<LEVEL>(const model_m_gram & gram);

Definition at line 504 of file generic_trie_base.hpp.

9.69.1.2 #define INSTANTIATE_TRIE_TEMPLATE_TYPE(*TRIE_TYPE_NAME, ...*)

Value:

```
template class TRIE_TYPE_NAME<__VA_ARGS__>; \
            INSTANTIATE_TRIE_FUNCS_LEVEL(M_GRAM_LEVEL_1, TRIE_TYPE_NAME, __VA_ARGS__); \
            INSTANTIATE_TRIE_FUNCS_LEVEL(M_GRAM_LEVEL_2, TRIE_TYPE_NAME, __VA_ARGS__); \
            INSTANTIATE_TRIE_FUNCS_LEVEL(M_GRAM_LEVEL_3, TRIE_TYPE_NAME, __VA_ARGS__); \
            INSTANTIATE_TRIE_FUNCS_LEVEL(M_GRAM_LEVEL_4, TRIE_TYPE_NAME, __VA_ARGS__); \
            INSTANTIATE_TRIE_FUNCS_LEVEL(M_GRAM_LEVEL_5, TRIE_TYPE_NAME, __VA_ARGS__); \
            INSTANTIATE_TRIE_FUNCS_LEVEL(M_GRAM_LEVEL_6, TRIE_TYPE_NAME, __VA_ARGS__); \
            INSTANTIATE_TRIE_FUNCS_LEVEL(M_GRAM_LEVEL_7, TRIE_TYPE_NAME, __VA_ARGS__);
```

Definition at line 507 of file generic_trie_base.hpp.

9.69.1.3 #define REPORT_COLLISION_WARNING(*gram, word_id, contextId, prevProb, prevBackOff, newProb, newBackOff*)

Value:

```
LOG_WARNING << "The " << gram.get_m_gram_level() << "-Gram : " << (string) gram            \
                      << " has been already seen! Word Id: " << SSTR(word_id)
        \
                      << ", context Id: " << SSTR(contextId) << ". "
        \
                      << "Changing the (prob,back-off) data from ("                                \
                      << prevProb << "," << prevBackOff << ") to ("                                \
                      << newProb << "," << newBackOff << ")" << END_LOG;
```

Definition at line 64 of file generic_trie_base.hpp.

9.70 inc/server/lm/models/h2d_map_trie.hpp File Reference

```
#include <string>
#include "server/lm/lm_consts.hpp"
#include "common/utils/exceptions.hpp"
#include "server/lm/dictionaries/aword_index.hpp"
#include "server/lm/dictionaries/hashing_word_index.hpp"
#include "server/lm/mgrams/model_m_gram.hpp"
#include "server/lm/mgrams/m_gram_id.hpp"
#include "common/utils/file/text_piece_reader.hpp"
#include "common/utils/containers/array_utils.hpp"
#include "common/utils/containers/fixed_size_hashmap.hpp"
#include "generic_trie_base.hpp"
```

Classes

- struct uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_GramData< TPayloadType >
- class uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::__H2DMapTrie

Typedefs

- typedef h2d_map_trie< basic_word_index > uva::smt::bpbd::server::lm::TH2DMapTrieBasic
- typedef h2d_map_trie< counting_word_index > uva::smt::bpbd::server::lm::TH2DMapTrieCount
- typedef h2d_map_trie< basic_optimizing_word_index > uva::smt::bpbd::server::lm::TH2DMapTrieOptBasic
- typedef h2d_map_trie< counting_optimizing_word_index > uva::smt::bpbd::server::lm::TH2DMapTrieOpt↩ Count
- typedef h2d_map_trie< hashing_word_index > uva::smt::bpbd::server::lm::TH2DMapTrieHashing

9.71 inc/server/lm/models/layered_trie_base.hpp File Reference

```
#include <string>
```

```
#include <cstring>
#include "server/lm/lm_consts.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/file/text_piece_reader.hpp"
#include "server/lm/dictionaries/basic_word_index.hpp"
#include "server/lm/dictionaries/counting_word_index.hpp"
#include "server/lm/dictionaries/optimizing_word_index.hpp"
#include "server/lm/models/generic_trie_base.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::layered_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE↩ _BUCKETS_FACTOR >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::__LayeredTrieBase

Macros

- #define LAYERED_BASE_ENSURE_CONTEXT(query, status)
- #define INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(CLASS_NAME, WORD_IDX_TYP↩ E) template class CLASS_NAME<WORD_IDX_TYPE >;

Functions

- template<typename TrieType , phrase_length CURR_LEVEL, bool GET_BACK_OFF_CTX_ID, debug_levels_enum LOG_LEVEL = debug_levels_enum::DEBUG1>
  phrase_length uva::smt::bpbd::server::lm::__LayeredTrieBase::search_m_gram_ctx_id (const TrieType &trie, const word_uid *const word_ids, TLongId &prev_ctx_id, TLongId &ctx_id)
- template<typename TrieType , phrase_length CURR_LEVEL, debug_levels_enum LOG_LEVEL>
  void uva::smt::bpbd::server::lm::__LayeredTrieBase::get_context_id (TrieType &trie, const model_m_gram &gram, TLongId &ctx_id)

9.71.1 Macro Definition Documentation

9.71.1.1 #define INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE(*CLASS_NAME, WORD_IDX_TYPE*) template class CLASS_NAME<WORD_IDX_TYPE >;

Definition at line 315 of file layered_trie_base.hpp.

9.71.1.2 #define LAYERED_BASE_ENSURE_CONTEXT(*query, status*)

Value:

```
if (query.get_curr_ctx_ref() == UNDEFINED_WORD_ID) { \
                BASE::ensure_context(query, status); \
            } else { \
                status = MGramStatusEnum::GOOD_PRESENT_MGS; \
            }
```

Definition at line 142 of file layered_trie_base.hpp.

9.72 inc/server/lm/models/m_gram_query.hpp File Reference

```
#include <string>
#include <ostream>
#include <algorithm>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "server/server_configs.hpp"
#include "server/lm/lm_consts.hpp"
#include "server/lm/mgrams/query_m_gram.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::m_gram_query

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

9.73 inc/server/lm/models/w2c_array_trie.hpp File Reference

```
#include <string>
#include <cstdlib>
#include <cmath>
#include <algorithm>
#include "server/lm/lm_consts.hpp"
#include "common/utils/logging/logger.hpp"
#include "layered_trie_base.hpp"
#include "server/lm/dictionaries/aword_index.hpp"
#include "server/lm/dictionaries/hashing_word_index.hpp"
#include "common/utils/containers/array_utils.hpp"
#include "common/utils/containers/dynamic_memory_arrays.hpp"
```

Classes

- struct uva::smt::bpbd::server::lm::__W2CArrayTrie::S_M_GramData< PAYLOAD_TYPE >
- class uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >
- class uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::WordDataEntry< ARRAY_ELEM_T↩ YPE >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::__W2CArrayTrie

Typedefs

- typedef S_M_GramData< m_gram_payload > uva::smt::bpbd::server::lm::__W2CArrayTrie::T_M_GramData
- typedef S_M_GramData< prob_weight > uva::smt::bpbd::server::lm::__W2CArrayTrie::T_N_GramData
- typedef w2c_array_trie< basic_word_index > uva::smt::bpbd::server::lm::TW2CArrayTrieBasic
- typedef w2c_array_trie< counting_word_index > uva::smt::bpbd::server::lm::TW2CArrayTrieCount
- typedef w2c_array_trie< basic_optimizing_word_index > uva::smt::bpbd::server::lm::TW2CArrayTrieOpt↩Basic
- typedef w2c_array_trie< counting_optimizing_word_index > uva::smt::bpbd::server::lm::TW2CArrayTrie↩OptCount
- typedef w2c_array_trie< hashing_word_index > uva::smt::bpbd::server::lm::TW2CArrayTrieHashing

Functions

- bool uva::smt::bpbd::server::lm::__W2CArrayTrie::operator< (const T_M_GramData &one, const T_M_↩GramData &two)
- bool uva::smt::bpbd::server::lm::__W2CArrayTrie::operator< (const T_N_GramData &one, const T_N_↩GramData &two)

9.74 inc/server/lm/models/w2c_hybrid_trie.hpp File Reference

```
#include <string>
#include "server/lm/lm_consts.hpp"
#include "common/utils/logging/logger.hpp"
#include "layered_trie_base.hpp"
#include "server/lm/dictionaries/aword_index.hpp"
#include "server/lm/dictionaries/hashing_word_index.hpp"
#include "server/lm/models/w2ch_um_storage.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

Typedefs

- typedef w2c_hybrid_trie< basic_word_index > uva::smt::bpbd::server::lm::TW2CHybridTrieBasic
- typedef w2c_hybrid_trie< counting_word_index > uva::smt::bpbd::server::lm::TW2CHybridTrieCount
- typedef w2c_hybrid_trie< basic_optimizing_word_index > uva::smt::bpbd::server::lm::TW2CHybridTrie↩ OptBasic
- typedef w2c_hybrid_trie< counting_optimizing_word_index > uva::smt::bpbd::server::lm::TW2CHybridTrie↩ OptCount
- typedef w2c_hybrid_trie< hashing_word_index > uva::smt::bpbd::server::lm::TW2CHybridTrieHashing

9.75 inc/server/lm/models/w2ch_um_storage.hpp File Reference

```
#include <inttypes.h>
#include <utility>
#include <unordered_map>
#include <map>
#include "server/lm/lm_consts.hpp"
#include "common/utils/logging/logger.hpp"
#include "server/lm/dictionaries/aword_index.hpp"
#include "common/utils/containers/greedy_memory_allocator.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::W2CH_UM_Storage
- class uva::smt::bpbd::server::lm::W2CH_UM_StorageFactory< N >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

Typedefs

- typedef pair< const TShortId, TShortId > uva::smt::bpbd::server::lm::TStorageMapEntry
- typedef greedy_memory_allocator< TStorageMapEntry > uva::smt::bpbd::server::lm::TStorageMap↩ Allocator
- typedef unordered_map< TShortId, TShortId, std::hash< TShortId >, std::equal_to< TShortId >, T↩ StorageMapAllocator > uva::smt::bpbd::server::lm::TStorageUnsignedMap
- typedef map< TShortId, TShortId > uva::smt::bpbd::server::lm::TStorageMap

9.76 inc/server/lm/models/word_index_trie_base.hpp File Reference

```
#include <string>
```

```
#include "server/lm/lm_consts.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/file/text_piece_reader.hpp"
#include "server/lm/dictionaries/basic_word_index.hpp"
#include "server/lm/dictionaries/counting_word_index.hpp"
#include "server/lm/dictionaries/optimizing_word_index.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::word_index_trie_base< WordIndex >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

9.77 inc/server/lm/proxy/lm_fast_query_proxy.hpp File Reference

```
#include "common/utils/file/text_piece_reader.hpp"
#include "server/server_configs.hpp"
#include "server/lm/lm_consts.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::proxy

9.78 inc/server/lm/proxy/lm_fast_query_proxy_local.hpp File Reference

```
#include <algorithm>
#include "server/lm/proxy/lm_fast_query_proxy.hpp"
#include "server/lm/models/m_gram_query.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::proxy

9.79 inc/server/lm/proxy/lm_proxy.hpp File Reference

```
#include "server/lm/proxy/lm_slow_query_proxy.hpp"
#include "server/lm/proxy/lm_fast_query_proxy.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::proxy::lm_proxy

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::proxy

9.80 inc/server/lm/proxy/lm_proxy_local.hpp File Reference

```
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/monitor/statistics_monitor.hpp"
#include "server/server_configs.hpp"
#include "server/lm/lm_configs.hpp"
#include "server/lm/lm_consts.hpp"
#include "server/lm/proxy/lm_fast_query_proxy.hpp"
#include "server/lm/proxy/lm_fast_query_proxy_local.hpp"
#include "server/lm/proxy/lm_slow_query_proxy.hpp"
#include "server/lm/proxy/lm_slow_query_proxy_local.hpp"
#include "server/lm/builders/lm_basic_builder.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::proxy::lm_proxy_local

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::proxy

9.81 inc/server/lm/proxy/lm_slow_query_proxy.hpp File Reference

```
#include "common/utils/file/text_piece_reader.hpp"
#include "server/server_configs.hpp"
#include "server/lm/lm_consts.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::proxy

9.82 inc/server/lm/proxy/lm_slow_query_proxy_local.hpp File Reference

```
#include <string>
#include "server/lm/proxy/lm_slow_query_proxy.hpp"
#include "server/lm/models/m_gram_query.hpp"
```

Classes

- class uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local< trie_type >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::proxy

9.83 inc/server/rm/builders/rm_basic_builder.hpp File Reference

```
#include <cmath>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/file/text_piece_reader.hpp"
#include "common/utils/string_utils.hpp"
#include "server/tm/tm_configurator.hpp"
#include "server/tm/proxy/tm_query_proxy.hpp"
#include "server/common/models/phrase_uid.hpp"
#include "server/rm/rm_parameters.hpp"
#include "server/rm/models/rm_entry.hpp"
```

Classes

- class uva::smt::bpbd::server::rm::builders::rm_basic_builder< model_type, reader_type >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::rm
- uva::smt::bpbd::server::rm::builders

9.84 inc/server/rm/models/rm_basic_model.hpp File Reference

```
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "server/rm/rm_consts.hpp"
#include "server/rm/models/rm_entry.hpp"
#include "server/rm/models/rm_query.hpp"
#include "common/utils/containers/fixed_size_hashmap.hpp"
```

Classes

- class uva::smt::bpbd::server::rm::models::rm_basic_model

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::rm
- uva::smt::bpbd::server::rm::models

9.85 inc/server/rm/models/rm_entry.hpp File Reference

```
#include <string>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "server/common/models/phrase_uid.hpp"
```

Classes

- class uva::smt::bpbd::server::rm::models::rm_entry_temp< num_features >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::rm
- uva::smt::bpbd::server::rm::models

Typedefs

- typedef rm_entry_temp< NUM_RM_FEATURES > uva::smt::bpbd::server::rm::models::rm_entry

Enumerations

- enum uva::smt::bpbd::server::rm::models::reordering_orientation {
  uva::smt::bpbd::server::rm::models::UNKNOWN_ORIENT = 0, uva::smt::bpbd::server::rm::models::MONO↩
  TONE_ORIENT = UNKNOWN_ORIENT + 1, uva::smt::bpbd::server::rm::models::SWAP_ORIENT = MON↩
  OTONE_ORIENT + 1, uva::smt::bpbd::server::rm::models::DISCONT_LEFT_ORIENT = SWAP_ORIENT +
  1,
  uva::smt::bpbd::server::rm::models::DISCONT_RIGHT_ORIENT = DISCONT_LEFT_ORIENT + 1, uva↩
  ::smt::bpbd::server::rm::models::size = DISCONT_RIGHT_ORIENT + 1 }

9.86 inc/server/rm/models/rm_query.hpp File Reference

```
#include <string>
#include <unordered_map>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "server/rm/models/rm_entry.hpp"
```

Classes

- class uva::smt::bpbd::server::rm::models::rm_query< model_type >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::rm
- uva::smt::bpbd::server::rm::models

9.87 inc/server/rm/proxy/rm_proxy.hpp File Reference

```
#include "server/rm/proxy/rm_query_proxy.hpp"
```

Classes

- class uva::smt::bpbd::server::rm::proxy::rm_proxy

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::rm
- uva::smt::bpbd::server::rm::proxy

9.88 inc/server/rm/proxy/rm_proxy_local.hpp File Reference

```
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/monitor/statistics_monitor.hpp"
#include "server/server_configs.hpp"
#include "server/rm/rm_configs.hpp"
#include "server/rm/proxy/rm_query_proxy.hpp"
#include "server/rm/proxy/rm_query_proxy_local.hpp"
```

Classes

- class uva::smt::bpbd::server::rm::proxy::rm_proxy_local

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::rm
- uva::smt::bpbd::server::rm::proxy

9.89 inc/server/rm/proxy/rm_query_proxy.hpp File Reference

```
#include <vector>
#include "server/common/models/phrase_uid.hpp"
#include "server/rm/models/rm_entry.hpp"
```

Classes

- class uva::smt::bpbd::server::rm::proxy::rm_query_proxy

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::rm
- uva::smt::bpbd::server::rm::proxy

9.90 inc/server/rm/proxy/rm_query_proxy_local.hpp File Reference

```
#include "server/rm/proxy/rm_query_proxy.hpp"
#include "server/rm/models/rm_entry.hpp"
#include "server/rm/models/rm_query.hpp"
```

Classes

- class uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::rm
- uva::smt::bpbd::server::rm::proxy

9.91 inc/server/rm/rm_configs.hpp File Reference

```
#include "server/server_configs.hpp"
#include "common/utils/file/cstyle_file_reader.hpp"
#include "common/utils/file/file_stream_reader.hpp"
#include "common/utils/file/memory_mapped_file_reader.hpp"
#include "server/rm/models/rm_basic_model.hpp"
#include "server/rm/builders/rm_basic_builder.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::rm

Typedefs

- typedef rm_basic_model uva::smt::bpbd::server::rm::rm_model_type
- typedef cstyle_file_reader uva::smt::bpbd::server::rm::rm_model_reader
- typedef rm_basic_builder< rm_model_type, rm_model_reader > uva::smt::bpbd::server::rm::rm_builder_type

9.92 inc/server/rm/rm_configurator.hpp File Reference

```
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "server/rm/rm_parameters.hpp"
#include "server/rm/proxy/rm_proxy.hpp"
#include "server/rm/proxy/rm_proxy_local.hpp"
#include "server/rm/proxy/rm_query_proxy.hpp"
```

Classes

- class uva::smt::bpbd::server::rm::rm_configurator

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::rm

9.93 inc/server/rm/rm_consts.hpp File Reference

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::rm
- uva::smt::bpbd::server::rm::models
- uva::smt::bpbd::server::rm::models::__rm_basic_model

9.94 inc/server/rm/rm_parameters.hpp File Reference

```
#include <string>
#include <ostream>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/string_utils.hpp"
#include "server/server_configs.hpp"
```

Classes

- struct uva::smt::bpbd::server::rm::rm_parameters

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::rm

9.95 inc/server/server_configs.hpp File Reference

```
#include "server_consts.hpp"
#include <string>
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::decoder
- uva::smt::bpbd::server::tm
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::rm

Macros

- #define SERVER_CONFIGS_HPP

Variables

- const prob_weight uva::smt::bpbd::server::lm::DEF_UNK_WORD_LOG_PROB_WEIGHT = -10.0f

9.95.1 Macro Definition Documentation

9.95.1.1 #define SERVER_CONFIGS_HPP

Definition at line 29 of file server_configs.hpp.

9.96 inc/server/server_consts.hpp File Reference

```
#include <string>
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::tm
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::rm

Typedefs

- typedef uint16_t uva::smt::bpbd::server::phrase_length
- typedef float uva::smt::bpbd::server::prob_weight
- typedef uint64_t uva::smt::bpbd::server::phrase_uid
- typedef uint64_t uva::smt::bpbd::server::word_uid

9.97 inc/server/server_parameters.hpp File Reference

```
#include <string>
#include <ostream>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "decoder/de_parameters.hpp"
#include "lm/lm_parameters.hpp"
#include "rm/rm_parameters.hpp"
#include "tm/tm_parameters.hpp"
```

Classes

- struct uva::smt::bpbd::server::server_parameters

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server

9.98 inc/server/tm/builders/tm_basic_builder.hpp File Reference

```
#include <cmath>
#include <unordered_map>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/file/text_piece_reader.hpp"
#include "common/utils/string_utils.hpp"
#include "server/server_consts.hpp"
#include "server/common/models/phrase_uid.hpp"
#include "server/lm/proxy/lm_fast_query_proxy.hpp"
#include "server/lm/lm_configurator.hpp"
#include "server/tm/tm_parameters.hpp"
#include "server/tm/models/tm_target_entry.hpp"
#include "server/tm/models/tm_source_entry.hpp"
```

Classes

- class uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::tm
- uva::smt::bpbd::server::tm::builders

Typedefs

- typedef unordered_map< phrase_uid, size_t > uva::smt::bpbd::server::tm::builders::sizes_map

9.99 inc/server/tm/models/tm_basic_model.hpp File Reference

```
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "server/lm/proxy/lm_fast_query_proxy.hpp"
#include "server/tm/tm_consts.hpp"
#include "server/tm/models/tm_source_entry.hpp"
#include "server/tm/models/tm_query.hpp"
#include "common/utils/containers/fixed_size_hashmap.hpp"
```

Classes

- class uva::smt::bpbd::server::tm::models::tm_basic_model

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::tm
- uva::smt::bpbd::server::tm::models

9.100 inc/server/tm/models/tm_query.hpp File Reference

```
#include <string>
#include <unordered_map>
#include <vector>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "server/tm/models/tm_source_entry.hpp"
```

Classes

- class uva::smt::bpbd::server::tm::models::tm_query< model_type >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::tm
- uva::smt::bpbd::server::tm::models

9.101 inc/server/tm/models/tm_source_entry.hpp File Reference

```
#include <string>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/containers/fixed_size_hashmap.hpp"
#include "server/common/models/phrase_uid.hpp"
#include "server/lm/proxy/lm_fast_query_proxy.hpp"
#include "server/tm/models/tm_target_entry.hpp"
```

Classes

- class uva::smt::bpbd::server::tm::models::tm_source_entry

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::tm
- uva::smt::bpbd::server::tm::models

Typedefs

- typedef const tm_source_entry uva::smt::bpbd::server::tm::models::tm_const_source_entry
- typedef tm_const_source_entry * uva::smt::bpbd::server::tm::models::tm_const_source_entry_ptr
- typedef tm_source_entry * uva::smt::bpbd::server::tm::models::tm_source_entry_ptr

9.102 inc/server/tm/models/tm_target_entry.hpp File Reference

```
#include <cstring>
#include <string>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/hashing_utils.hpp"
#include "server/lm/proxy/lm_fast_query_proxy.hpp"
#include "server/common/models/phrase_uid.hpp"
```

Classes

- class uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::tm
- uva::smt::bpbd::server::tm::models

Typedefs

- typedef tm_target_entry_temp< NUM_TM_FEATURES > uva::smt::bpbd::server::tm::models::tm_target_↩ entry
- typedef const tm_target_entry uva::smt::bpbd::server::tm::models::tm_const_target_entry
- typedef prob_weight uva::smt::bpbd::server::tm::models::feature_array[tm_target_entry::NUM_FEATURES]

9.103 inc/server/tm/proxy/tm_proxy.hpp File Reference

```
#include "server/tm/proxy/tm_query_proxy.hpp"
```

Classes

- class uva::smt::bpbd::server::tm::proxy::tm_proxy

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::tm
- uva::smt::bpbd::server::tm::proxy

9.104 inc/server/tm/proxy/tm_proxy_local.hpp File Reference

```
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/monitor/statistics_monitor.hpp"
#include "server/tm/tm_configs.hpp"
#include "server/tm/proxy/tm_query_proxy.hpp"
#include "server/tm/proxy/tm_query_proxy_local.hpp"
#include "server/tm/builders/tm_basic_builder.hpp"
```

Classes

- class uva::smt::bpbd::server::tm::proxy::tm_proxy_local

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::tm
- uva::smt::bpbd::server::tm::proxy

9.105 inc/server/tm/proxy/tm_query_proxy.hpp File Reference

```
#include "server/tm/tm_configs.hpp"
#include "server/tm/models/tm_source_entry.hpp"
```

Classes

- class uva::smt::bpbd::server::tm::proxy::tm_query_proxy

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::tm
- uva::smt::bpbd::server::tm::proxy

9.106 inc/server/tm/proxy/tm_query_proxy_local.hpp File Reference

```
#include "server/tm/proxy/tm_query_proxy.hpp"
#include "server/tm/models/tm_query.hpp"
```

Classes

- class uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local< model_type >

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::tm
- uva::smt::bpbd::server::tm::proxy

9.107 inc/server/tm/tm_configs.hpp File Reference

```
#include "server/server_configs.hpp"
#include "common/utils/file/cstyle_file_reader.hpp"
#include "common/utils/file/file_stream_reader.hpp"
#include "common/utils/file/memory_mapped_file_reader.hpp"
#include "server/tm/models/tm_basic_model.hpp"
#include "server/tm/builders/tm_basic_builder.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::tm

Typedefs

- typedef tm_basic_model uva::smt::bpbd::server::tm::tm_model_type
- typedef cstyle_file_reader uva::smt::bpbd::server::tm::tm_model_reader
- typedef tm_basic_builder< tm_model_type, tm_model_reader > uva::smt::bpbd::server::tm::tm_builder_type

9.108 inc/server/tm/tm_configurator.hpp File Reference

```
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "server/tm/tm_parameters.hpp"
#include "server/tm/proxy/tm_proxy.hpp"
#include "server/tm/proxy/tm_proxy_local.hpp"
#include "server/tm/proxy/tm_query_proxy.hpp"
```

Classes

- class uva::smt::bpbd::server::tm::tm_configurator

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::tm

9.109 inc/server/tm/tm_consts.hpp File Reference

```
#include <string>
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::tm
- uva::smt::bpbd::server::tm::models
- uva::smt::bpbd::server::tm::models::__tm_basic_model

9.110 inc/server/tm/tm_parameters.hpp File Reference

```
#include <string>
#include <ostream>
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/string_utils.hpp"
#include "server/server_configs.hpp"
```

Classes

- struct uva::smt::bpbd::server::tm::tm_parameters

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::tm

9.111 inc/server/trans_job_pool.hpp File Reference

```
#include <map>
#include <vector>
#include <websocketpp/server.hpp>
#include "trans_task_pool.hpp"
#include "common/messaging/trans_session_id.hpp"
#include "common/messaging/trans_job_id.hpp"
#include "common/messaging/trans_job_request.hpp"
#include "common/utils/threads.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "trans_task_id.hpp"
#include "trans_job.hpp"
```

Classes

- class uva::smt::bpbd::server::trans_job_pool

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server

Macros

- #define TRANS_JOB_POOL_HPP

9.111.1 Macro Definition Documentation

9.111.1.1 #define TRANS_JOB_POOL_HPP

Definition at line 51 of file trans_job_pool.hpp.

9.112 inc/server/trans_task.hpp File Reference

```
#include "common/utils/threads.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/messaging/trans_session_id.hpp"
#include "common/messaging/trans_job_id.hpp"
#include "common/messaging/trans_job_code.hpp"
#include "trans_task_id.hpp"
#include "server/decoder/de_configurator.hpp"
#include "server/decoder/sentence/sentence_decoder.hpp"
```

Classes

- class uva::smt::bpbd::server::trans_task

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server

Typedefs

- typedef trans_task * uva::smt::bpbd::server::trans_task_ptr

9.113 inc/server/trans_task_id.hpp File Reference

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::task_id

Typedefs

- typedef uint64_t uva::smt::bpbd::server::task_id_type

9.114 inc/server/trans_task_pool.hpp File Reference

```
#include <vector>
#include <deque>
#include "trans_task.hpp"
#include "common/utils/threads.hpp"
#include "server/trans_task_pool_worker.hpp"
```

Classes

- class uva::smt::bpbd::server::trans_task_pool

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server

9.115 inc/server/trans_task_pool_worker.hpp File Reference

```
#include "trans_task.hpp"
#include "common/utils/threads.hpp"
```

Classes

- class uva::smt::bpbd::server::trans_task_pool_worker

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server

9.116 inc/server/translation_server.hpp File Reference

```
#include <iostream>
#include <functional>
#include <websocketpp/config/asio_no_tls.hpp>
#include <websocketpp/server.hpp>
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/messaging/trans_job_response.hpp"
#include "common/messaging/trans_job_request.hpp"
#include "common/messaging/trans_job_code.hpp"
#include "trans_manager.hpp"
```

Classes

- class uva::smt::bpbd::server::translation_server

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server

Macros

- #define ASIO_STANDALONE

9.116.1 Macro Definition Documentation

9.116.1.1 #define ASIO_STANDALONE

Definition at line 32 of file translation_server.hpp.

9.117 README.md File Reference

9.118 src/client/bpbd_client.cpp File Reference

```
#include <string>
#include "tclap/CmdLine.h"
#include "main.hpp"
#include "client/client_config.hpp"
#include "client/trans_manager.hpp"
#include "common/utils/exceptions.hpp"
```

Macros

- #define PROGRAM_VERSION_STR "1.0"

Functions

- void create_arguments_parser ()
- void destroy_arguments_parser ()
- int main (int argc, char **argv)

9.118.1 Macro Definition Documentation

9.118.1.1 #define PROGRAM_VERSION_STR "1.0"

Definition at line 48 of file bpbd_client.cpp.

9.118.2 Function Documentation

9.118.2.1 void create_arguments_parser ( )

Creates and sets up the command line parameters parser

Definition at line 75 of file bpbd_client.cpp.

9.118.2.2 void destroy_arguments_parser ( )

Allows to deallocate the parameters parser if it is needed

Definition at line 115 of file bpbd_client.cpp.

9.118.2.3 int main (int *argc*, char ** *argv*)

The main program entry point

Definition at line 171 of file bpbd_client.cpp.

9.119 src/client/trans_job_status.cpp File Reference

```
#include "client/trans_job_status.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::client

Macros

- #define STATUS_UNKNOWN_STR "unknown"
- #define STATUS_UNDEFINED_STR "undefined"
- #define STATUS_REQ_INITIALIZED_STR "not-sent"
- #define STATUS_REQ_SENT_GOOD_STR "not-replied"
- #define STATUS_REQ_SENT_FAIL_STR "send-failed"
- #define STATUS_RES_RECEIVED_STR "replied"

Functions

- ostream & uva::smt::bpbd::client::operator<< (ostream &os, const trans_job_status &status)

9.119.1 Macro Definition Documentation

9.119.1.1 #define STATUS_REQ_INITIALIZED_STR "not-sent"

Definition at line 36 of file trans_job_status.cpp.

9.119.1.2 #define STATUS_REQ_SENT_FAIL_STR "send-failed"

Definition at line 38 of file trans_job_status.cpp.

9.119.1.3 #define STATUS_REQ_SENT_GOOD_STR "not-replied"

Definition at line 37 of file trans_job_status.cpp.

9.119.1.4 #define STATUS_RES_RECEIVED_STR "replied"

Definition at line 39 of file trans_job_status.cpp.

9.119.1.5 #define STATUS_UNDEFINED_STR "undefined"

Definition at line 35 of file trans_job_status.cpp.

9.119.1.6 #define STATUS_UNKNOWN_STR "unknown"

Definition at line 34 of file trans_job_status.cpp.

9.120 src/common/messaging/trans_job_code.cpp File Reference

```
#include "common/messaging/trans_job_code.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::common
- uva::smt::bpbd::common::messaging

Macros

- #define RESULT_UNKNOWN_STR "unknown"
- #define RESULT_UNDEFINED_STR "undefined"
- #define RESULT_OK_STR "good"
- #define RESULT_ERROR_STR "error"
- #define RESULT_CANCELED_STR "canceled"
- #define RESULT_PARTIAL_STR "partial"

Functions

- ostream & uva::smt::bpbd::common::messaging::operator<< (ostream &os, const trans_job_code &code)

9.120.1 Macro Definition Documentation

9.120.1.1 #define RESULT_CANCELED_STR "canceled"

Definition at line 39 of file trans_job_code.cpp.

9.120.1.2 #define RESULT_ERROR_STR "error"

Definition at line 38 of file trans_job_code.cpp.

9.120.1.3 #define RESULT_OK_STR "good"

Definition at line 37 of file trans_job_code.cpp.

9.120.1.4 #define RESULT_PARTIAL_STR "partial"

Definition at line 40 of file trans_job_code.cpp.

9.120.1.5 #define RESULT_UNDEFINED_STR "undefined"

Definition at line 36 of file trans_job_code.cpp.

9.120.1.6 #define RESULT_UNKNOWN_STR "unknown"

Definition at line 35 of file trans_job_code.cpp.

9.121 src/common/utils/logging/logger.cpp File Reference

```
#include <algorithm>
#include "common/utils/logging/logger.hpp"
#include "common/utils/monitor/statistics_monitor.hpp"
```

Namespaces

- uva
- uva::utils
- uva::utils::logging

Macros

- #define IS_ENOUGH_LOGGING_LEVEL(level) ((PROGRESS_ACTIVE_LEVEL <= LOGER_M_GRAM_↩ LEVEL_MAX) && (PROGRESS_ACTIVE_LEVEL <= level))

Functions

- std::ostream & uva::utils::logging::operator<< (std::ostream &stream, const unsigned char &value)
- std::ostream & uva::utils::logging::operator<< (std::ostream &stream, const signed char &value)

9.121.1 Macro Definition Documentation

9.121.1.1 #define IS_ENOUGH_LOGGING_LEVEL(*level*) ((PROGRESS_ACTIVE_LEVEL <= LOGER_M_GRAM_LEVEL_MAX) && (PROGRESS_ACTIVE_LEVEL <= level))

Definition at line 184 of file logger.cpp.

9.122 src/common/utils/monitor/statistics_monitor.cpp File Reference

```
#include "common/utils/monitor/statistics_monitor.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/logging/logger.hpp"
#include <cstring>
#include <cstdlib>
#include <cstdio>
#include <sstream>
```

Namespaces

- uva
- uva::utils
- uva::utils::monitor

9.123 src/server/bpbd_server.cpp File Reference

```
#include <cctype>
#include <cstdlib>
#include <string>
#include <websocketpp/common/thread.hpp>
#include <tclap/CmdLine.h>
#include "main.hpp"
#include "server/server_parameters.hpp"
#include "server/translation_server.hpp"
#include "common/utils/exceptions.hpp"
#include "server/decoder/de_configurator.hpp"
#include "server/lm/lm_configurator.hpp"
#include "server/tm/tm_configurator.hpp"
#include "server/rm/rm_configurator.hpp"
#include "server/cmd_line_handler.hpp"
```

Macros

- #define PROGRAM_VERSION_STR "1.0"

Functions

- void create_arguments_parser ()
- void destroy_arguments_parser ()
- void connect_to_models (const server_parameters &params)
- void disconnect_from_models ()
- int main (int argc, char **argv)

9.123.1 Macro Definition Documentation

9.123.1.1 #define PROGRAM_VERSION_STR "1.0"

Definition at line 62 of file bpbd_server.cpp.

9.123.2 Function Documentation

9.123.2.1 void connect_to_models (const server_parameters & *params*)

Allows to establish connections to the models: language, translation, reordering

Parameters

params	the parameters needed to establish connections to the models

Definition at line 201 of file bpbd_server.cpp.

9.123.2.2 void create_arguments_parser ( )

Creates and sets up the command line parameters parser

Definition at line 81 of file bpbd_server.cpp.

9.123.2.3 void destroy_arguments_parser ( )

Allows to deallocate the parameters parser if it is needed

Definition at line 97 of file bpbd_server.cpp.

9.123.2.4 void disconnect_from_models ( )

Allows to disconnect from the models: language, translation, reordering

Definition at line 218 of file bpbd_server.cpp.

9.123.2.5 int main (int *argc*, char ** *argv*)

The main program entry point

Definition at line 235 of file bpbd_server.cpp.

9.124 src/server/decoder/de_configurator.cpp File Reference

```
#include "server/decoder/de_configurator.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::decoder

9.125 src/server/lm/builders/lm_basic_builder.cpp File Reference

```
#include <iostream>
```

```
#include <string>
#include "common/utils/logging/logger.hpp"
#include "common/utils/string_utils.hpp"
#include "common/utils/file/cstyle_file_reader.hpp"
#include "common/utils/file/file_stream_reader.hpp"
#include "common/utils/file/memory_mapped_file_reader.hpp"
#include "server/lm/lm_consts.hpp"
#include "server/lm/builders/lm_basic_builder.hpp"
#include "server/lm/builders/lm_gram_builder_factory.hpp"
#include "server/lm/models/c2d_map_trie.hpp"
#include "server/lm/models/w2c_hybrid_trie.hpp"
#include "server/lm/models/c2w_array_trie.hpp"
#include "server/lm/models/w2c_array_trie.hpp"
#include "server/lm/models/c2d_hybrid_trie.hpp"
#include "server/lm/models/g2d_map_trie.hpp"
#include "server/lm/models/h2d_map_trie.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::arpa

Macros

- #define INSTANTIATE_TRIE_BUILDER_FILE_READER(TFileReaderModel)

Functions

- uva::smt::bpbd::server::lm::arpa::INSTANTIATE_TRIE_BUILDER_FILE_READER (cstyle_file_reader)
- uva::smt::bpbd::server::lm::arpa::INSTANTIATE_TRIE_BUILDER_FILE_READER (file_stream_reader)
- uva::smt::bpbd::server::lm::arpa::INSTANTIATE_TRIE_BUILDER_FILE_READER (memory_mapped_file_reader)

9.125.1 Macro Definition Documentation

9.125.1.1 #define INSTANTIATE_TRIE_BUILDER_FILE_READER(*TFileReaderModel*)

Definition at line 494 of file lm_basic_builder.cpp.

9.126 src/server/lm/builders/lm_gram_builder.cpp File Reference

```
#include "server/lm/builders/lm_gram_builder.hpp"
#include <string>
#include <vector>
#include <stdexcept>
#include <algorithm>
#include "common/utils/logging/logger.hpp"
#include "common/utils/string_utils.hpp"
#include "common/utils/exceptions.hpp"
#include "server/lm/dictionaries/basic_word_index.hpp"
#include "server/lm/dictionaries/counting_word_index.hpp"
#include "server/lm/dictionaries/optimizing_word_index.hpp"
#include "server/lm/dictionaries/hashing_word_index.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::arpa

Macros

- #define INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL_WEIGHT(LEVEL, IS_MULT_WEIGHT)
- #define INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL(LEVEL)

Functions

- uva::smt::bpbd::server::lm::arpa::INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_1)
- uva::smt::bpbd::server::lm::arpa::INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_2)
- uva::smt::bpbd::server::lm::arpa::INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_3)
- uva::smt::bpbd::server::lm::arpa::INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_4)
- uva::smt::bpbd::server::lm::arpa::INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_5)
- uva::smt::bpbd::server::lm::arpa::INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_6)
- uva::smt::bpbd::server::lm::arpa::INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL (M_GRAM_LEVEL_7)

9.126.1 Macro Definition Documentation

9.126.1.1 #define INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL(*LEVEL*)

Value:

```
INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL_WEIGHT(LEVEL, true); \
                INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL_WEIGHT(LEVEL, false);
```

Definition at line 193 of file lm_gram_builder.cpp.

9.126.1.2 #define INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL_WEIGHT(*LEVEL*, *IS_MULT_WEIGHT*)

Value:

```
template class lm_gram_builder<basic_word_index, LEVEL, IS_MULT_WEIGHT>; \
               template class lm_gram_builder<counting_word_index, LEVEL, IS_MULT_WEIGHT>; \
               template class lm_gram_builder<hashing_word_index, LEVEL, IS_MULT_WEIGHT>; \
               template class lm_gram_builder<basic_optimizing_word_index, LEVEL, IS_MULT_WEIGHT>; \
               template class lm_gram_builder<counting_optimizing_word_index, LEVEL, IS_MULT_WEIGHT>;
```

Definition at line 186 of file lm_gram_builder.cpp.

9.127 src/server/lm/lm_configurator.cpp File Reference

```
#include "server/lm/lm_configurator.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

9.128 src/server/lm/lm_query.cpp File Reference

```
#include <string>
#include <iostream>
#include <sstream>
#include <fstream>
#include <math.h>
#include "tclap/CmdLine.h"
#include "main.hpp"
#include "common/utils/monitor/statistics_monitor.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/string_utils.hpp"
#include "common/utils/exceptions.hpp"
#include "common/utils/file/afile_reader.hpp"
#include "common/utils/file/memory_mapped_file_reader.hpp"
#include "common/utils/file/file_stream_reader.hpp"
#include "common/utils/file/cstyle_file_reader.hpp"
#include "server/lm/lm_executor.hpp"
```

Macros

- #define PROGRAM_VERSION_STR "1.1"

Functions

- void create_arguments_parser ()
- void destroy_arguments_parser ()
- int main (int argc, char **argv)

9.128.1 Macro Definition Documentation

9.128.1.1 #define PROGRAM_VERSION_STR "1.1"

Definition at line 61 of file lm_query.cpp.

9.128.2 Function Documentation

9.128.2.1 void create_arguments_parser ( )

Creates and sets up the command line parameters parser

Definition at line 83 of file lm_query.cpp.

9.128.2.2 void destroy_arguments_parser ( )

Allows to deallocate the parameters parser if it is needed

Definition at line 105 of file lm_query.cpp.

9.128.2.3 int main (int *argc*, char ** *argv*)

The main program entry point

Definition at line 149 of file lm_query.cpp.

9.129 src/server/lm/mgrams/byte_m_gram_id.cpp File Reference

```
#include "server/lm/mgrams/m_gram_id.hpp"
#include <inttypes.h>
#include "server/lm/lm_consts.hpp"
#include "server/lm/lm_configs.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "server/lm/dictionaries/aword_index.hpp"
#include "common/utils/file/text_piece_reader.hpp"
#include "common/utils/hashing_utils.hpp"
#include "common/utils/math_utils.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::m_grams
- uva::smt::bpbd::server::lm::m_grams::m_gram_id

Macros

- #define MAX_VALUE_IN_BYTES(NUM_BYTES) (const_expr::power(2, BYTES_TO_BITS(NUM_BYTES)) - 1)

9.129.1 Macro Definition Documentation

9.129.1.1 #define MAX_VALUE_IN_BYTES(*NUM_BYTES*) (const_expr::power(2, BYTES_TO_BITS(NUM_BYTES)) - 1)

Definition at line 59 of file byte_m_gram_id.cpp.

9.130 src/server/lm/mgrams/model_m_gram.cpp File Reference

```
#include "server/lm/mgrams/model_m_gram.hpp"
#include "common/utils/string_utils.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::m_grams

Functions

- ostream & uva::smt::bpbd::server::lm::m_grams::operator<< (ostream &stream, const model_m_gram &gram)

9.131 src/server/lm/mgrams/query_m_gram.cpp File Reference

```
#include "server/lm/mgrams/query_m_gram.hpp"
#include "common/utils/string_utils.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm
- uva::smt::bpbd::server::lm::m_grams

Functions

- ostream & uva::smt::bpbd::server::lm::m_grams::operator<< (ostream &stream, const query_m_gram &gram)

9.132 src/server/lm/models/c2d_hybrid_trie.cpp File Reference

```
#include "server/lm/models/c2d_hybrid_trie.hpp"
#include <stdexcept>
#include <sstream>
#include <algorithm>
#include "common/utils/logging/logger.hpp"
#include "common/utils/string_utils.hpp"
#include "server/lm/dictionaries/basic_word_index.hpp"
#include "server/lm/dictionaries/counting_word_index.hpp"
#include "server/lm/dictionaries/optimizing_word_index.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

Functions

- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_hybrid_trie, basic_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_hybrid_trie, counting_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_hybrid_trie, hashing_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_hybrid_trie, basic_optimizing_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_hybrid_trie, counting_optimizing_word_index)

9.133 src/server/lm/models/c2d_map_trie.cpp File Reference

```
#include "server/lm/models/c2d_map_trie.hpp"
#include <stdexcept>
#include <sstream>
#include <algorithm>
#include "common/utils/logging/logger.hpp"
#include "common/utils/string_utils.hpp"
#include "server/lm/dictionaries/basic_word_index.hpp"
#include "server/lm/dictionaries/counting_word_index.hpp"
#include "server/lm/dictionaries/optimizing_word_index.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

Functions

- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_map_trie, basic_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_map_trie, counting_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_map_trie, hashing_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_map_trie, basic_optimizing_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2d_map_trie, counting_optimizing_word_index)

9.134 src/server/lm/models/c2w_array_trie.cpp File Reference

```
#include "server/lm/models/c2w_array_trie.hpp"
#include <inttypes.h>
#include "server/lm/lm_consts.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "server/lm/dictionaries/basic_word_index.hpp"
#include "server/lm/dictionaries/counting_word_index.hpp"
#include "server/lm/dictionaries/optimizing_word_index.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

Functions

- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2w_array_trie, basic_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2w_array_trie, counting_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2w_array_trie, hashing_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2w_array_trie, basic_optimizing_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (c2w_array_trie, counting_optimizing_word_index)

9.135 src/server/lm/models/g2d_map_trie.cpp File Reference

```
#include "server/lm/models/g2d_map_trie.hpp"
```

```
#include <inttypes.h>
#include <algorithm>
#include "server/lm/lm_consts.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "server/lm/dictionaries/basic_word_index.hpp"
#include "server/lm/dictionaries/counting_word_index.hpp"
#include "server/lm/dictionaries/optimizing_word_index.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

Functions

- uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (g2d_map_trie, basic_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (g2d_map_trie, counting_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (g2d_map_trie, hashing_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (g2d_map_trie, basic_optimizing_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (g2d_map_trie, counting_optimizing_word_index)

9.136 src/server/lm/models/h2d_map_trie.cpp File Reference

```
#include "server/lm/models/h2d_map_trie.hpp"
#include <inttypes.h>
#include <algorithm>
#include "server/lm/lm_consts.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "server/lm/dictionaries/basic_word_index.hpp"
#include "server/lm/dictionaries/counting_word_index.hpp"
#include "server/lm/dictionaries/optimizing_word_index.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

Functions

- uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (h2d_map_trie, basic_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (h2d_map_trie, counting_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (h2d_map_trie, hashing_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (h2d_map_trie, basic_optimizing_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_TRIE_TEMPLATE_TYPE (h2d_map_trie, counting_optimizing_word_index)

9.137 src/server/lm/models/m_gram_query.cpp File Reference

```
#include "server/lm/models/m_gram_query.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

Functions

- ostream & uva::smt::bpbd::server::lm::operator<< (ostream &stream, const m_gram_query &query)

9.138 src/server/lm/models/w2c_array_trie.cpp File Reference

```
#include "server/lm/models/w2c_array_trie.hpp"
#include <inttypes.h>
#include "server/lm/lm_consts.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "server/lm/dictionaries/basic_word_index.hpp"
#include "server/lm/dictionaries/counting_word_index.hpp"
#include "server/lm/dictionaries/optimizing_word_index.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

Functions

- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_array_trie, basic_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_array_trie, counting_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_array_trie, hashing_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_array_trie, basic_optimizing_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_array_trie, counting_optimizing_word_index)

9.139 src/server/lm/models/w2c_hybrid_trie.cpp File Reference

```
#include "server/lm/models/w2c_hybrid_trie.hpp"
#include <inttypes.h>
#include "server/lm/lm_consts.hpp"
#include "common/utils/logging/logger.hpp"
#include "common/utils/exceptions.hpp"
#include "server/lm/dictionaries/basic_word_index.hpp"
#include "server/lm/dictionaries/counting_word_index.hpp"
#include "server/lm/dictionaries/optimizing_word_index.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::lm

Functions

- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_hybrid_trie, basic_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_hybrid_trie, counting_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_hybrid_trie, hashing_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_hybrid_trie, basic_optimizing_word_index)
- uva::smt::bpbd::server::lm::INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE (w2c_hybrid_trie, counting_optimizing_word_index)

9.140 src/server/rm/rm_configurator.cpp File Reference

```
#include "server/rm/rm_configurator.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::rm

9.141 src/server/tm/models/tm_target_entry.cpp File Reference

```
#include "server/tm/models/tm_target_entry.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::tm
- uva::smt::bpbd::server::tm::models

9.142 src/server/tm/tm_configurator.cpp File Reference

```
#include "server/tm/tm_configurator.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server
- uva::smt::bpbd::server::tm

9.143 src/server/trans_task_pool.cpp File Reference

```
#include <functional>
#include "server/trans_task_pool.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server

9.144 src/server/trans_task_pool_worker.cpp File Reference

```
#include "server/trans_task_pool_worker.hpp"
#include "server/trans_task_pool.hpp"
```

Namespaces

- uva
- uva::smt
- uva::smt::bpbd
- uva::smt::bpbd::server

Index

__FILENAME__
logger.hpp, 406
__attribute__
uva::smt::bpbd::server::lm, 57
uva::smt::bpbd::server::lm::dictionary::__optimizing_word_index, 65
_allocBytes
uva::utils::containers::greedy_memory_storage, 169
_manager
uva::utils::containers::alloc::greedy_memory_allocator, 166
_memIncTypesEnumStr
uva::utils::containers, 73
_memoryBuffers
uva::utils::containers::greedy_memory_storage, 169
_numBytes
uva::utils::containers::greedy_memory_storage, 169
_pBuffer
uva::utils::containers::greedy_memory_storage, 169
~BitmapHashCache
uva::smt::bpbd::server::lm::caching::BitmapHashCache, 98
~S_M_GramData
uva::smt::bpbd::server::lm::__G2DMapTrie::S_M_GramData, 264
uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_GramData, 262
~W2CH_UM_Storage
uva::smt::bpbd::server::lm::W2CH_UM_Storage, 379
~W2CH_UM_StorageFactory
uva::smt::bpbd::server::lm::W2CH_UM_StorageFactory, 380
~afile_reader
uva::utils::file::afile_reader, 88
~aword_index
uva::smt::bpbd::server::lm::dictionary::aword_index, 90
~basic_word_index
uva::smt::bpbd::server::lm::dictionary::basic_word_index, 94
~c2d_hybrid_trie
uva::smt::bpbd::server::lm::c2d_hybrid_trie, 116
~c2d_map_trie
uva::smt::bpbd::server::lm::c2d_map_trie, 120
~c2w_array_trie
uva::smt::bpbd::server::lm::c2w_array_trie, 124
~circular_queue
uva::utils::containers::circular_queue, 127
~cstyle_file_reader
uva::utils::file::cstyle_file_reader, 135
~dynamic_stack_array
uva::utils::containers::dynamic_stack_array, 142
~file_stream_reader
uva::utils::file::file_stream_reader, 146
~fixed_size_hashmap
uva::utils::containers::fixed_size_hashmap, 150
~g2d_map_trie
uva::smt::bpbd::server::lm::g2d_map_trie, 152
~generic_trie_base
uva::smt::bpbd::server::lm::generic_trie_base, 156
~greedy_memory_allocator
uva::utils::containers::alloc::greedy_memory_allocator, 163
~greedy_memory_storage
uva::utils::containers::greedy_memory_storage, 168
~h2d_map_trie
uva::smt::bpbd::server::lm::h2d_map_trie, 171
~hashing_word_index
uva::smt::bpbd::server::lm::dictionary::hashing_word_index, 173
~lm_basic_builder
uva::smt::bpbd::server::lm::arpa::lm_basic_builder, 181
~lm_fast_query_proxy
uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy, 184
~lm_fast_query_proxy_local
uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local, 188
~lm_gram_builder
uva::smt::bpbd::server::lm::arpa::lm_gram_builder, 191
~lm_gram_builder_factory
uva::smt::bpbd::server::lm::arpa::lm_gram_builder_factory, 194
~lm_proxy
uva::smt::bpbd::server::lm::proxy::lm_proxy, 197
~lm_proxy_local
uva::smt::bpbd::server::lm::proxy::lm_proxy_local, 199
~lm_slow_query_proxy
uva::smt::bpbd::server::lm::proxy::lm_slow_

query_proxy, 202
~lm_slow_query_proxy_local
uva::smt::bpbd::server::lm::proxy::lm_slow_↩
query_proxy_local, 203
~logger
uva::utils::logging::logger, 206
~multi_stack
uva::smt::bpbd::server::decoder::stack::multi_↩
stack, 226
~optimizing_word_index
uva::smt::bpbd::server::lm::dictionary::optimizing↩
_word_index, 228
~phrase_data_entry
uva::smt::bpbd::server::decoder::sentence↩
::phrase_data_entry, 236
~rm_basic_model
uva::smt::bpbd::server::rm::models::rm_basic_↩
model, 242
~rm_entry_temp
uva::smt::bpbd::server::rm::models::rm_entry_↩
temp, 249
~rm_proxy
uva::smt::bpbd::server::rm::proxy::rm_proxy, 253
~rm_proxy_local
uva::smt::bpbd::server::rm::proxy::rm_proxy_local,
254
~rm_query
uva::smt::bpbd::server::rm::models::rm_query, 256
~rm_query_proxy
uva::smt::bpbd::server::rm::proxy::rm_query_↩
proxy, 257
~rm_query_proxy_local
uva::smt::bpbd::server::rm::proxy::rm_query_↩
proxy_local, 259
~sentence_decoder
uva::smt::bpbd::server::decoder::sentence↩
::sentence_decoder, 266
~stack_level
uva::smt::bpbd::server::decoder::stack::stack_↩
level, 271
~stack_state_templ
uva::smt::bpbd::server::decoder::stack::stack_↩
state_templ, 277
~tm_basic_builder
uva::smt::bpbd::server::tm::builders::tm_basic_↩
builder, 298
~tm_basic_model
uva::smt::bpbd::server::tm::models::tm_basic_↩
model, 302
~tm_proxy
uva::smt::bpbd::server::tm::proxy::tm_proxy, 309
~tm_proxy_local
uva::smt::bpbd::server::tm::proxy::tm_proxy_local,
311
~tm_query
uva::smt::bpbd::server::tm::models::tm_query, 313
~tm_query_proxy
uva::smt::bpbd::server::tm::proxy::tm_query_proxy,
314
~tm_query_proxy_local
uva::smt::bpbd::server::tm::proxy::tm_query_↩
proxy_local, 316
~tm_source_entry
uva::smt::bpbd::server::tm::models::tm_source_↩
entry, 317
~tm_target_entry_temp
uva::smt::bpbd::server::tm::models::tm_target_↩
entry_temp, 321
~trans_job
uva::smt::bpbd::client::trans_job, 325
uva::smt::bpbd::server::trans_job, 327
~trans_job_pool
uva::smt::bpbd::server::trans_job_pool, 333
~trans_manager
uva::smt::bpbd::client::trans_manager, 349
uva::smt::bpbd::server::trans_manager, 346
~trans_task
uva::smt::bpbd::server::trans_task, 353
~trans_task_pool
uva::smt::bpbd::server::trans_task_pool, 356
~trans_task_pool_worker
uva::smt::bpbd::server::trans_task_pool_worker,
358
~translation_client
uva::smt::bpbd::client::translation_client, 359
~upp_diag_matrix
uva::utils::containers::upp_diag_matrix, 367
~uva_exception
uva::utils::exceptions::uva_exception, 369
~w2c_array_trie
uva::smt::bpbd::server::lm::w2c_array_trie, 371
~w2c_hybrid_trie
uva::smt::bpbd::server::lm::w2c_hybrid_trie, 376

A
hashing_utils.hpp, 404
a_bool_flag
uva::utils::threads, 85
ASCII_SPACE_CHAR
uva::utils::text, 84
ASIO_STANDALONE
translation_client.hpp, 390
translation_server.hpp, 461
ASSERT_CONDITION_THROW
exceptions.hpp, 400
ASSERT_SANITY_THROW
exceptions.hpp, 400
acr_bool_flag
uva::utils::threads, 85
add_before
uva::smt::bpbd::server::decoder::stack::stack_↩
level, 271
add_entry
uva::smt::bpbd::server::rm::models::rm_basic_↩
model, 243
add_job
uva::smt::bpbd::server::trans_job_pool, 333

add_last
uva::smt::bpbd::server::decoder::stack::stack_↩ level, 273
add_m_gram
uva::smt::bpbd::server::lm::c2d_hybrid_trie, 116
uva::smt::bpbd::server::lm::c2d_map_trie, 120
uva::smt::bpbd::server::lm::c2w_array_trie, 124
uva::smt::bpbd::server::lm::g2d_map_trie, 153
uva::smt::bpbd::server::lm::generic_trie_base, 156
uva::smt::bpbd::server::lm::h2d_map_trie, 171
uva::smt::bpbd::server::lm::w2c_array_trie, 372
uva::smt::bpbd::server::lm::w2c_hybrid_trie, 376
add_new_element
uva::utils::containers::fixed_size_hashmap, 150
add_new_state_function
uva::smt::bpbd::server::decoder::stack, 47
add_stack_state
uva::smt::bpbd::server::decoder::stack::multi_↩ stack, 227
add_state
uva::smt::bpbd::server::decoder::stack::stack_↩ level, 273
add_target
uva::smt::bpbd::server::tm::models::tm_source_↩ entry, 318
add_unk_translation
uva::smt::bpbd::server::tm::builders::tm_basic_↩ builder, 299
address
uva::utils::containers::alloc::greedy_memory_↩ allocator, 164
afile_reader
uva::utils::file::afile_reader, 88
allocate
uva::utils::containers::alloc::greedy_memory_↩ allocator, 164
uva::utils::containers::dynamic_stack_array, 142
uva::utils::containers::greedy_memory_storage, 168
allocate_byte_m_gram_id
uva::smt::bpbd::server::lm::m_grams::m_gram_↩ id::Byte_M_Gram_Id, 103
allocate_container
uva::utils::containers::alloc, 74
allocate_decoder
uva::smt::bpbd::server::decoder::de_configurator, 137
allocate_fast_query_proxy
uva::smt::bpbd::server::lm::lm_configurator, 182
uva::smt::bpbd::server::lm::proxy::lm_proxy, 197
uva::smt::bpbd::server::lm::proxy::lm_proxy_local, 199
allocate_query_proxy
uva::smt::bpbd::server::rm::proxy::rm_proxy, 253
uva::smt::bpbd::server::rm::proxy::rm_proxy_local, 254
uva::smt::bpbd::server::rm::rm_configurator, 247
uva::smt::bpbd::server::tm::proxy::tm_proxy, 309
uva::smt::bpbd::server::tm::proxy::tm_proxy_local, 311
uva::smt::bpbd::server::tm::tm_configurator, 306
allocate_slow_query_proxy
uva::smt::bpbd::server::lm::lm_configurator, 182
uva::smt::bpbd::server::lm::proxy::lm_proxy, 197
uva::smt::bpbd::server::lm::proxy::lm_proxy_local, 199
array_utils.hpp
BSEARCH_ONE_FIELD, 394
BSEARCH_TWO_FIELDS, 394
DECLARE_STATIC_BSEARCH_ID_FIELD_CO↩ MPARE_FUNC, 394
at
uva::smt::bpbd::server::lm::W2CH_UM_Storage, 379
available
uva::utils::containers::alloc::greedy_memory_↩ allocator, 164

B
hashing_utils.hpp, 404
BAD_END_WORD_UNKNOWN_MGS
uva::smt::bpbd::server::lm, 55
BAD_NO_PAYLOAD_MGS
uva::smt::bpbd::server::lm, 55
BASE
uva::smt::bpbd::server::lm::c2d_hybrid_trie, 116
uva::smt::bpbd::server::lm::c2d_map_trie, 119
uva::smt::bpbd::server::lm::c2w_array_trie, 123
uva::smt::bpbd::server::lm::g2d_map_trie, 152
uva::smt::bpbd::server::lm::generic_trie_base, 155
uva::smt::bpbd::server::lm::h2d_map_trie, 170
uva::smt::bpbd::server::lm::layered_trie_base, 177
uva::smt::bpbd::server::lm::m_grams::model_m_↩ gram, 224
uva::smt::bpbd::server::lm::m_grams::query_m_↩ gram, 238
uva::smt::bpbd::server::lm::w2c_array_trie, 371
uva::smt::bpbd::server::lm::w2c_hybrid_trie, 375
BEGIN_SENT_TAG_UID
uva::smt::bpbd::server::rm::models::rm_basic_↩ model, 247
BSEARCH_ONE_FIELD
array_utils.hpp, 394
BSEARCH_TWO_FIELDS
array_utils.hpp, 394
BYTE_IDX
math_utils.hpp, 411
BYTE_M_GRAM_ID_TABLES_HPP
m_gram_id_tables.hpp, 430
BYTES_ONE_MB
uva::utils::monitor, 84
BYTES_TO_BITS
math_utils.hpp, 411
basic_optimizing_word_index
uva::smt::bpbd::server::lm::dictionary, 64
basic_word_index

uva::smt::bpbd::server::lm::dictionary::basic_↩
word_index, 94
begin
uva::smt::bpbd::server::lm::dictionary::basic_↩
word_index, 95
uva::smt::bpbd::server::tm::models::tm_source_↩
entry, 318
begin_entry
uva::smt::bpbd::server::tm::models::tm_basic_↩
model, 302
begin_idx
uva::smt::bpbd::server::lm::c2w_array_trie::T↩
SubArrReference, 364
begins_with
uva::smt::bpbd::server, 43
BitmapHashCache
uva::smt::bpbd::server::lm::caching::BitmapHash↩
Cache, 98
bpbd_client.cpp
create_arguments_parser, 461
destroy_arguments_parser, 461
main, 461
PROGRAM_VERSION_STR, 461
bpbd_server.cpp
connect_to_models, 466
create_arguments_parser, 466
destroy_arguments_parser, 466
disconnect_from_models, 466
main, 466
PROGRAM_VERSION_STR, 465
build
uva::smt::bpbd::server::lm::arpa::lm_basic_builder,
181
uva::smt::bpbd::server::rm::builders::rm_basic_↩
builder, 240
uva::smt::bpbd::server::tm::builders::tm_basic_↩
builder, 299
byte_m_gram_id.cpp
MAX_VALUE_IN_BYTES, 471

C
hashing_utils.hpp, 404
c2d_hybrid_trie
uva::smt::bpbd::server::lm::c2d_hybrid_trie, 116
c2d_map_trie
uva::smt::bpbd::server::lm::c2d_map_trie, 119
c2w_array_trie
uva::smt::bpbd::server::lm::c2w_array_trie, 123
CONSTANT
uva::utils::containers, 73
cache_m_gram_hash
uva::smt::bpbd::server::lm::caching::BitmapHash↩
Cache, 98
cancel
uva::smt::bpbd::server::trans_job, 327
uva::smt::bpbd::server::trans_task, 353
cancel_all_jobs
uva::smt::bpbd::server::trans_job_pool, 333
cancel_jobs
uva::smt::bpbd::server::trans_job_pool, 333
cancel_task_notifier
uva::smt::bpbd::server::trans_task, 351
ceil
uva::utils::math::const_expr, 83
check_jobs_done_and_notify
uva::smt::bpbd::client::trans_manager, 349
cio
uva::smt::bpbd::server::lm::w2c_array_trie::Word↩
DataEntry, 385
circular_queue
uva::utils::containers::circular_queue, 127
client
uva::smt::bpbd::client::translation_client, 359
close
uva::utils::file::afile_reader, 88
uva::utils::file::cstyle_file_reader, 135
uva::utils::file::file_stream_reader, 146
uva::utils::file::memory_mapped_file_reader, 221
close_session
uva::smt::bpbd::server::trans_manager, 346
combine_job_result
uva::smt::bpbd::server::trans_job, 327
compare
uva::smt::bpbd::server::lm::__C2WArrayTrie, 58
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 103
compute_futue_costs
uva::smt::bpbd::server::decoder::sentence↩
::sentence_decoder, 266
compute_m_gram_id
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 105
conn_close_notifier
uva::smt::bpbd::client::translation_client, 359
connect
uva::smt::bpbd::client::translation_client, 360
uva::smt::bpbd::server::decoder::de_configurator,
137
uva::smt::bpbd::server::lm::lm_configurator, 182
uva::smt::bpbd::server::lm::proxy::lm_proxy, 197
uva::smt::bpbd::server::lm::proxy::lm_proxy_local,
200
uva::smt::bpbd::server::rm::proxy::rm_proxy, 253
uva::smt::bpbd::server::rm::proxy::rm_proxy_local,
254
uva::smt::bpbd::server::rm::rm_configurator, 248
uva::smt::bpbd::server::tm::proxy::tm_proxy, 309
uva::smt::bpbd::server::tm::proxy::tm_proxy_local,
311
uva::smt::bpbd::server::tm::tm_configurator, 306
connect_to_models
bpbd_server.cpp, 466
const_iterator
uva::smt::bpbd::server::lm::W2CH_UM_Storage,
379
const_pointer

uva::utils::containers::alloc::greedy_memory_↩
allocator, 162
const_reference
uva::utils::containers::alloc::greedy_memory_↩
allocator, 162
construct
uva::utils::containers::alloc::greedy_memory_↩
allocator, 164
copy_string
uva::utils::file::text_piece_reader, 292
count_and_prune
uva::smt::bpbd::server::decoder::stack::stack_↩
state_templ, 277
count_source_phrases
uva::smt::bpbd::server::tm::builders::tm_basic_↩
builder, 299
count_source_target_phrases
uva::smt::bpbd::server::rm::builders::rm_basic_↩
builder, 240
count_word
uva::smt::bpbd::server::lm::dictionary::aword_↩
index, 90
uva::smt::bpbd::server::lm::dictionary::basic_↩
word_index, 95
uva::smt::bpbd::server::lm::dictionary::counting_↩
word_index, 133
uva::smt::bpbd::server::lm::dictionary::optimizing↩
_word_index, 228
count_words
uva::smt::bpbd::server::decoder::sentence↩
::sentence_decoder, 266
counting_optimizing_word_index
uva::smt::bpbd::server::lm::dictionary, 64
counting_word_index
uva::smt::bpbd::server::lm::dictionary::counting_↩
word_index, 132
covered_info
uva::smt::bpbd::server::decoder::stack::state_↩
data_templ, 284
covered_to_string
uva::smt::bpbd::server::decoder::stack::state_↩
data_templ, 286
create
uva::smt::bpbd::server::lm::W2CH_UM_Storage↩
Factory, 380
create_arguments_parser
bpbd_client.cpp, 461
bpbd_server.cpp, 466
lm_query.cpp, 470
create_m_gram_id
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 105
create_phrase_id
uva::smt::bpbd::server::lm::m_grams::phrase_↩
base, 232
cstyle_file_reader
uva::utils::file::cstyle_file_reader, 135
ctx_id
uva::smt::bpbd::server::lm::__C2WArrayTrie::T↩
CtxIdProbData, 290
cut_the_tail
uva::smt::bpbd::server::decoder::stack::stack_↩
state_templ, 279
cwfold
hashing_utils.hpp, 404
cwmixa
hashing_utils.hpp, 405
cwmixb
hashing_utils.hpp, 405

DEBUG
uva::utils::logging, 82
DEBUG1
uva::utils::logging, 82
DEBUG1_PARAM_VALUE
logger.hpp, 406
DEBUG2
uva::utils::logging, 82
DEBUG2_PARAM_VALUE
logger.hpp, 407
DEBUG3
uva::utils::logging, 82
DEBUG3_PARAM_VALUE
logger.hpp, 407
DEBUG4
uva::utils::logging, 82
DEBUG4_PARAM_VALUE
logger.hpp, 407
DEBUG_PARAM_VALUE
logger.hpp, 407
DECLARE_STACK_GRAM_ID
m_gram_id.hpp, 429
DECLARE_STATIC_BSEARCH_ID_FIELD_COMPA↩
RE_FUNC
array_utils.hpp, 394
DEF_UNK_WORD_LOG_PROB_WEIGHT
uva::smt::bpbd::server::lm, 57
DISCONT_LEFT_ORIENT
uva::smt::bpbd::server::rm::models, 69
DISCONT_RIGHT_ORIENT
uva::smt::bpbd::server::rm::models, 69
DO_SANITY_CHECKS
uva::utils::exceptions, 80
data
uva::utils::containers::dynamic_stack_array, 142
de_parameters
uva::smt::bpbd::server::decoder, 47
de_parameters_struct
uva::smt::bpbd::server::decoder::de_parameters↩
_struct, 138
de_serialize
uva::smt::bpbd::common::messaging::trans_job↩
_request, 337
uva::smt::bpbd::common::messaging::trans_job↩
_response, 340
deallocate

uva::utils::containers::alloc::greedy_memory_↩
allocator, 166
deallocate_container
uva::utils::containers::alloc, 74
debug_levels_enum
uva::utils::logging, 82
delete_job
uva::smt::bpbd::server::trans_job_pool, 334
destroy
uva::utils::containers::alloc::greedy_memory_↩
allocator, 166
destroy_arguments_parser
bpbd_client.cpp, 461
bpbd_server.cpp, 466
lm_query.cpp, 470
difference_type
uva::utils::containers::alloc::greedy_memory_↩
allocator, 162
disconnect
uva::smt::bpbd::client::translation_client, 360
uva::smt::bpbd::server::decoder::de_configurator, 137
uva::smt::bpbd::server::lm::lm_configurator, 183
uva::smt::bpbd::server::lm::proxy::lm_proxy, 198
uva::smt::bpbd::server::lm::proxy::lm_proxy_local, 200
uva::smt::bpbd::server::rm::proxy::rm_proxy, 253
uva::smt::bpbd::server::rm::proxy::rm_proxy_local, 255
uva::smt::bpbd::server::rm::rm_configurator, 248
uva::smt::bpbd::server::tm::proxy::tm_proxy, 309
uva::smt::bpbd::server::tm::proxy::tm_proxy_local, 311
uva::smt::bpbd::server::tm::tm_configurator, 307
disconnect_from_models
bpbd_server.cpp, 466
dispose_decoder
uva::smt::bpbd::server::decoder::de_configurator, 137
dispose_fast_query_proxy
uva::smt::bpbd::server::lm::lm_configurator, 183
uva::smt::bpbd::server::lm::proxy::lm_proxy, 198
uva::smt::bpbd::server::lm::proxy::lm_proxy_local, 200
dispose_query_proxy
uva::smt::bpbd::server::rm::proxy::rm_proxy, 253
uva::smt::bpbd::server::rm::proxy::rm_proxy_local, 255
uva::smt::bpbd::server::rm::rm_configurator, 248
uva::smt::bpbd::server::tm::proxy::tm_proxy, 310
uva::smt::bpbd::server::tm::proxy::tm_proxy_local, 311
uva::smt::bpbd::server::tm::tm_configurator, 307
dispose_slow_query_proxy
uva::smt::bpbd::server::lm::lm_configurator, 183
uva::smt::bpbd::server::lm::proxy::lm_proxy, 198
uva::smt::bpbd::server::lm::proxy::lm_proxy_local, 200
do_post_actions
uva::smt::bpbd::server::lm::dictionary::aword_↩
index, 90
uva::smt::bpbd::server::lm::dictionary::basic_↩
word_index, 95
uva::smt::bpbd::server::lm::dictionary::counting_↩
word_index, 133
uva::smt::bpbd::server::lm::dictionary::optimizing↩
_word_index, 229
do_post_word_count
uva::smt::bpbd::server::lm::dictionary::aword_↩
index, 91
uva::smt::bpbd::server::lm::dictionary::basic_↩
word_index, 95
uva::smt::bpbd::server::lm::dictionary::counting_↩
word_index, 133
uva::smt::bpbd::server::lm::dictionary::optimizing↩
_word_index, 229
done_job_notifier
uva::smt::bpbd::server::trans_job, 326
done_task_notifier
uva::smt::bpbd::server::trans_task, 351
dynamic_memory_arrays.hpp
EXTRACT_C, 397
EXTRACT_P, 397
EXTRACT_PC, 397
EXTRACT_PCS, 397
EXTRACT_PS, 397
EXTRACT_S, 397
dynamic_stack_array
uva::utils::containers::dynamic_stack_array, 142

ELEMENT_TYPE_PTR
uva::utils::containers::dynamic_stack_array, 141
END_LOG
logger.hpp, 407
END_SENT_TAG_UID
uva::smt::bpbd::server::rm::models::rm_basic_↩
model, 247
ERROR
uva::utils::logging, 82
ERROR_PARAM_VALUE
logger.hpp, 407
EXTRACT_C
dynamic_memory_arrays.hpp, 397
EXTRACT_P
dynamic_memory_arrays.hpp, 397
EXTRACT_PC
dynamic_memory_arrays.hpp, 397
EXTRACT_PCS
dynamic_memory_arrays.hpp, 397
EXTRACT_PS
dynamic_memory_arrays.hpp, 397
EXTRACT_S
dynamic_memory_arrays.hpp, 397
element_type_ptr
uva::utils::containers::upp_diag_matrix, 367
empty_queue
uva::utils::containers::circular_queue, 127

end
uva::smt::bpbd::server::lm::W2CH_UM_Storage, 379
uva::smt::bpbd::server::lm::dictionary::basic_word_index, 95
end_idx
uva::smt::bpbd::server::lm::c2w_array_trie::TSubArrReference, 365
ensure_context
uva::smt::bpbd::server::lm::layered_trie_base, 179
exceptions.hpp
ASSERT_CONDITION_THROW, 400
ASSERT_SANITY_THROW, 400
THROW_EXCEPTION, 400
THROW_MUST_NOT_CALL, 401
THROW_MUST_OVERRIDE, 401
THROW_NOT_IMPLEMENTED, 401
execute
uva::smt::bpbd::server::lm::generic_trie_base, 156
uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy, 185
uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local, 188
uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy, 202
uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local, 203
uva::smt::bpbd::server::rm::models::rm_query, 256
uva::smt::bpbd::server::rm::proxy::rm_query_proxy, 258
uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local, 260
uva::smt::bpbd::server::tm::models::tm_query, 313
uva::smt::bpbd::server::tm::proxy::tm_query_proxy, 314
uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local, 316
expand
uva::smt::bpbd::server::decoder::stack::multi_stack, 227
uva::smt::bpbd::server::decoder::stack::stack_level, 273
uva::smt::bpbd::server::decoder::stack::stack_state_templ, 279
expand_left
uva::smt::bpbd::server::decoder::stack::stack_state_templ, 279
expand_length
uva::smt::bpbd::server::decoder::stack::stack_state_templ, 279
expand_length_if_not_covered
uva::smt::bpbd::server::decoder::stack::stack_state_templ, 279
expand_right
uva::smt::bpbd::server::decoder::stack::stack_state_templ, 280
expand_trans
uva::smt::bpbd::server::decoder::stack::stack_state_templ, 280

FIRST_VALID_CTX_ID
uva::smt::bpbd::server::lm::generic_trie_base, 160
feature_array
uva::smt::bpbd::server::tm::models, 71
file_stream_reader
uva::utils::file::file_stream_reader, 146
finalize
uva::smt::bpbd::server::decoder::de_parameters_struct, 139
uva::smt::bpbd::server::lm::lm_parameters, 195
uva::smt::bpbd::server::rm::rm_parameters, 252
uva::smt::bpbd::server::tm::models::tm_basic_model, 302
uva::smt::bpbd::server::tm::models::tm_source_entry, 318
uva::smt::bpbd::server::tm::tm_parameters, 308
finalize_entry
uva::smt::bpbd::server::tm::models::tm_basic_model, 302
find
uva::smt::bpbd::server::lm::W2CH_UM_Storage, 379
find_begin_end_entries
uva::smt::bpbd::server::rm::models::rm_basic_model, 244
find_first_subseq
uva::utils::file::text_piece_reader, 292
find_recombine
uva::smt::bpbd::server::decoder::stack::stack_level, 273
find_unk_entry
uva::smt::bpbd::server::rm::models::rm_basic_model, 244
finished_job_notifier
uva::smt::bpbd::server::trans_job_pool, 332
fixed_size_hashmap
uva::utils::containers::fixed_size_hashmap, 148
func
uva::smt::bpbd::server::lm::arpa::TAddGramFunct, 289
func_ptr
uva::utils::containers::ELEMENT_DEALLOC_FUNC, 145
uva::utils::containers::utils::T_IS_COMPARE_FUNC, 289
func_type
uva::utils::containers::ELEMENT_DEALLOC_FUNC, 145
uva::utils::containers::utils::T_IS_COMPARE_FUNC, 289
future_cost
uva::smt::bpbd::server::decoder::sentence::phrase_data_entry, 236

g2d_map_trie
uva::smt::bpbd::server::lm::g2d_map_trie, 152
GET_ASSERT

main.hpp, 414
GOOD_PRESENT_MGS
uva::smt::bpbd::server::lm, 55
generic_trie_base
uva::smt::bpbd::server::lm::generic_trie_base, 156
generic_trie_base.hpp
INSTANTIATE_TRIE_FUNCS_LEVEL, 436
INSTANTIATE_TRIE_TEMPLATE_TYPE, 436
REPORT_COLLISION_WARNING, 436
get
uva::utils::logging::logger, 206, 207
get16bits
hashing_utils.hpp, 405
get_begin_c_str
uva::utils::file::text_piece_reader, 292
get_begin_ptr
uva::utils::file::text_piece_reader, 292
get_begin_tag_entry
uva::smt::bpbd::server::rm::models::rm_basic_↩
model, 244
get_begin_tag_reordering
uva::smt::bpbd::server::rm::proxy::rm_query_↩
proxy, 258
uva::smt::bpbd::server::rm::proxy::rm_query_↩
proxy_local, 260
get_begin_tag_uid
uva::smt::bpbd::server::lm::proxy::lm_fast_query↩
_proxy, 185
uva::smt::bpbd::server::lm::proxy::lm_fast_query↩
_proxy_local, 188
get_best_trans
uva::smt::bpbd::server::decoder::stack::multi_↩
stack, 227
uva::smt::bpbd::server::decoder::stack::stack_↩
level, 273
get_builder
uva::smt::bpbd::server::lm::arpa::lm_gram_↩
builder_factory, 194
get_cached_context_id
uva::smt::bpbd::server::lm::layered_trie_base, 179
get_capacity
uva::utils::containers::circular_queue, 128
get_code
uva::smt::bpbd::common::messaging::trans_job↩
_response, 340
uva::smt::bpbd::server::trans_job, 327
uva::smt::bpbd::server::trans_task, 353
get_context_id
uva::smt::bpbd::server::lm::__LayeredTrieBase, 60
get_cpu_time
uva::utils::monitor::stat_monitor, 283
get_ctx_id
uva::smt::bpbd::server::lm::c2d_hybrid_trie, 117
uva::smt::bpbd::server::lm::c2d_map_trie, 120
uva::smt::bpbd::server::lm::c2w_array_trie, 124
uva::smt::bpbd::server::lm::layered_trie_base, 179
uva::smt::bpbd::server::lm::w2c_array_trie, 372
uva::smt::bpbd::server::lm::w2c_hybrid_trie, 376
get_curr_begin_word_id
uva::smt::bpbd::server::lm::m_gram_query, 213
get_curr_ctx_ref
uva::smt::bpbd::server::lm::m_gram_query, 213
get_curr_end_word_id
uva::smt::bpbd::server::lm::m_gram_query, 213
get_curr_level
uva::smt::bpbd::server::lm::m_gram_query, 213
get_curr_level_m1
uva::smt::bpbd::server::lm::m_gram_query, 213
get_curr_level_m2
uva::smt::bpbd::server::lm::m_gram_query, 213
get_curr_level_str
uva::utils::logging::logger, 207
get_curr_m_gram_hash
uva::smt::bpbd::server::lm::m_gram_query, 214
get_curr_m_gram_id
uva::smt::bpbd::server::lm::m_gram_query, 214
get_curr_payload_ref
uva::smt::bpbd::server::lm::m_gram_query, 214
get_curr_uni_gram_word_id
uva::smt::bpbd::server::lm::m_gram_query, 214
get_dim
uva::utils::containers::upp_diag_matrix, 367
get_element
uva::utils::containers::fixed_size_hashmap, 150
get_elems
uva::utils::containers::circular_queue, 128
get_end_tag_entry
uva::smt::bpbd::server::rm::models::rm_basic_↩
model, 244
get_end_tag_reordering
uva::smt::bpbd::server::rm::proxy::rm_query_↩
proxy, 258
uva::smt::bpbd::server::rm::proxy::rm_query_↩
proxy_local, 260
get_end_tag_uid
uva::smt::bpbd::server::lm::proxy::lm_fast_query↩
_proxy, 185
uva::smt::bpbd::server::lm::proxy::lm_fast_query↩
_proxy_local, 188
get_entry
uva::smt::bpbd::server::rm::models::rm_basic_↩
model, 244, 245
get_first
uva::utils::file::afile_reader, 88
uva::utils::file::text_piece_reader, 292
get_first_line
uva::utils::file::afile_reader, 88
uva::utils::file::cstyle_file_reader, 135
uva::utils::file::file_stream_reader, 146
uva::utils::file::memory_mapped_file_reader, 221
uva::utils::file::text_piece_reader, 293
get_first_space
uva::utils::file::afile_reader, 88
uva::utils::file::text_piece_reader, 293
get_first_tab
uva::utils::file::afile_reader, 88

uva::utils::file::text_piece_reader, 293
get_first_word_idx
uva::smt::bpbd::server::lm::m_grams::phrase_↩ base, 234
get_float
uva::smt::bpbd::common, 38
get_float_value
uva::smt::bpbd::server, 44
get_hash
uva::smt::bpbd::server::lm::m_grams::model_m_↩ gram, 224
uva::smt::bpbd::server::lm::m_grams::query_m_↩ gram, 238
get_int_value
uva::smt::bpbd::server, 44
get_integer
uva::smt::bpbd::common, 38
get_job_id
uva::smt::bpbd::common::messaging::trans_job↩ _request, 337
uva::smt::bpbd::common::messaging::trans_job↩ _response, 340
uva::smt::bpbd::server::trans_job, 327
get_last
uva::utils::file::afile_reader, 88
uva::utils::file::text_piece_reader, 293
get_last_space
uva::utils::file::afile_reader, 88
uva::utils::file::text_piece_reader, 295
get_last_word_id
uva::smt::bpbd::server::lm::m_grams::phrase_↩ base, 234
get_last_word_idx
uva::smt::bpbd::server::lm::m_grams::phrase_↩ base, 234
get_lm_weight
uva::smt::bpbd::server::lm::lm_parameters, 195
get_m_gram_payload
uva::smt::bpbd::server::lm::c2d_hybrid_trie, 117
uva::smt::bpbd::server::lm::c2d_map_trie, 120
uva::smt::bpbd::server::lm::c2w_array_trie, 124
uva::smt::bpbd::server::lm::g2d_map_trie, 153
uva::smt::bpbd::server::lm::generic_trie_base, 156
uva::smt::bpbd::server::lm::h2d_map_trie, 171
uva::smt::bpbd::server::lm::w2c_array_trie, 372
uva::smt::bpbd::server::lm::w2c_hybrid_trie, 377
get_m_gram_str
uva::smt::bpbd::server::lm::proxy::lm_fast_query↩ _proxy_local, 189
uva::smt::bpbd::server::lm::proxy::lm_slow_↩ query_proxy_local, 205
get_mem_incr_strat
uva::utils::containers, 73
get_mem_stat
uva::utils::monitor::stat_monitor, 283
get_message
uva::utils::exceptions::uva_exception, 369
get_min_cost
uva::smt::bpbd::server::tm::models::tm_source_↩ entry, 318
get_min_id
uva::smt::bpbd::common::messaging::id_manager, 176
get_n_gram_payload
uva::smt::bpbd::server::lm::c2d_hybrid_trie, 117
uva::smt::bpbd::server::lm::c2d_map_trie, 121
uva::smt::bpbd::server::lm::c2w_array_trie, 124
uva::smt::bpbd::server::lm::g2d_map_trie, 153
uva::smt::bpbd::server::lm::generic_trie_base, 158
uva::smt::bpbd::server::lm::h2d_map_trie, 171
uva::smt::bpbd::server::lm::w2c_array_trie, 372
uva::smt::bpbd::server::lm::w2c_hybrid_trie, 377
get_new_capacity
uva::utils::containers::mem_increase_strategy, 219
get_next_id
uva::smt::bpbd::common::messaging::id_manager, 176
get_next_new_token
uva::smt::bpbd::server::lm::m_grams::model_m_↩ gram, 224
get_num_of_sentences
uva::smt::bpbd::client::trans_manager, 349
get_num_words
uva::smt::bpbd::server::lm::m_grams::phrase_↩ base, 234
uva::smt::bpbd::server::tm::models::tm_target_↩ entry_temp, 322
get_number_of_words
uva::smt::bpbd::server::lm::dictionary::aword_↩ index, 91
uva::smt::bpbd::server::lm::dictionary::basic_↩ word_index, 95
uva::smt::bpbd::server::lm::dictionary::hashing_↩ word_index, 174
uva::smt::bpbd::server::lm::dictionary::optimizing↩ _word_index, 229
get_phrase_id_ref
uva::smt::bpbd::server::lm::m_grams::phrase_↩ base, 234
get_query_begin_word_idx
uva::smt::bpbd::server::lm::m_gram_query, 214
get_query_end_word_idx
uva::smt::bpbd::server::lm::m_gram_query, 215
get_query_str
uva::smt::bpbd::server::lm::proxy::lm_fast_query↩ _proxy_local, 189
uva::smt::bpbd::server::lm::proxy::lm_slow_↩ query_proxy_local, 205
get_reordering
uva::smt::bpbd::server::rm::models::rm_query, 257
uva::smt::bpbd::server::rm::proxy::rm_query_↩ proxy, 258
uva::smt::bpbd::server::rm::proxy::rm_query_↩ proxy_local, 260
get_report_interm_results
uva::smt::bpbd::server::lm::proxy::lm_fast_query↩

_proxy_local, 189
uva::smt::bpbd::server::lm::proxy::lm_slow_↩
query_proxy_local, 205
get_reporting_level
uva::utils::logging::logger, 207
get_reporting_levels
uva::utils::logging::logger, 207
get_rest_c_str
uva::utils::file::text_piece_reader, 295
get_rest_str
uva::utils::file::text_piece_reader, 295
get_session_id
uva::smt::bpbd::common::messaging::trans_job↩
_request, 337
uva::smt::bpbd::server::trans_job, 327
get_size
uva::smt::bpbd::server::decoder::stack::stack_↩
level, 274
uva::utils::containers::circular_queue, 128
get_source_entry
uva::smt::bpbd::server::tm::models::tm_basic_↩
model, 302
uva::smt::bpbd::server::tm::models::tm_query, 313
uva::smt::bpbd::server::tm::proxy::tm_query_proxy,
315
uva::smt::bpbd::server::tm::proxy::tm_query_↩
proxy_local, 316
get_source_lang
uva::smt::bpbd::common::messaging::trans_job↩
_request, 338
get_source_text
uva::smt::bpbd::server::trans_task, 353
get_source_uid
uva::smt::bpbd::server::tm::models::tm_source_↩
entry, 318
get_st_uid
uva::smt::bpbd::server::tm::models::tm_target_↩
entry_temp, 322
get_st_uids
uva::smt::bpbd::server::tm::models::tm_query, 313
uva::smt::bpbd::server::tm::models::tm_source_↩
entry, 319
uva::smt::bpbd::server::tm::proxy::tm_query_proxy,
315
uva::smt::bpbd::server::tm::proxy::tm_query_↩
proxy_local, 316
get_stack_level
uva::smt::bpbd::server::decoder::stack::stack_↩
state_templ, 280
get_strategy_info
uva::utils::containers::mem_increase_strategy, 219
get_string
uva::smt::bpbd::common, 38
get_string_value
uva::smt::bpbd::server, 44
get_t_c_s
uva::smt::bpbd::server::tm::models::tm_target_↩
entry_temp, 322
get_target_lang
uva::smt::bpbd::common::messaging::trans_job↩
_request, 338
get_target_phrase
uva::smt::bpbd::server::tm::models::tm_target_↩
entry_temp, 322
get_target_text
uva::smt::bpbd::server::trans_task, 353
get_targets
uva::smt::bpbd::server::tm::models::tm_source_↩
entry, 319
get_task_id
uva::smt::bpbd::server::trans_task, 353
get_tasks
uva::smt::bpbd::server::trans_job, 327
get_text
uva::smt::bpbd::common::messaging::trans_job↩
_request, 338
uva::smt::bpbd::common::messaging::trans_job↩
_response, 341
uva::smt::bpbd::server::trans_job, 328
get_total_weight
uva::smt::bpbd::server::tm::models::tm_target_↩
entry_temp, 322
get_translation
uva::smt::bpbd::server::decoder::stack::stack_↩
state_templ, 280
get_unigram_payload
uva::smt::bpbd::server::lm::c2d_hybrid_trie, 117
uva::smt::bpbd::server::lm::c2d_map_trie, 121
uva::smt::bpbd::server::lm::c2w_array_trie, 125
uva::smt::bpbd::server::lm::g2d_map_trie, 153
uva::smt::bpbd::server::lm::generic_trie_base, 158
uva::smt::bpbd::server::lm::h2d_map_trie, 172
uva::smt::bpbd::server::lm::w2c_array_trie, 372
uva::smt::bpbd::server::lm::w2c_hybrid_trie, 377
get_unk_word_prob
uva::smt::bpbd::server::lm::c2d_hybrid_trie, 117
uva::smt::bpbd::server::lm::c2d_map_trie, 121
uva::smt::bpbd::server::lm::c2w_array_trie, 125
uva::smt::bpbd::server::lm::g2d_map_trie, 153
uva::smt::bpbd::server::lm::generic_trie_base, 158
uva::smt::bpbd::server::lm::h2d_map_trie, 172
uva::smt::bpbd::server::lm::proxy::lm_fast_query↩
_proxy, 186
uva::smt::bpbd::server::lm::proxy::lm_fast_query↩
_proxy_local, 189
uva::smt::bpbd::server::lm::w2c_array_trie, 373
uva::smt::bpbd::server::lm::w2c_hybrid_trie, 377
get_uri
uva::smt::bpbd::client::translation_client, 360
get_weight
uva::smt::bpbd::server::rm::models::rm_entry_↩
temp, 249
get_weights
uva::smt::bpbd::server::rm::models::rm_entry_↩
temp, 249
get_word_id

uva::smt::bpbd::server::lm::dictionary::aword_index, 91
uva::smt::bpbd::server::lm::dictionary::basic_word_index, 96
uva::smt::bpbd::server::lm::dictionary::hashing_word_index, 174
uva::smt::bpbd::server::lm::dictionary::optimizing_word_index, 229
get_word_ids
uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy, 186
uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local, 190
uva::smt::bpbd::server::tm::models::tm_target_entry_temp, 323
get_word_index
uva::smt::bpbd::server::lm::word_index_trie_base, 383
getAvailableBytes
uva::utils::containers::greedy_memory_storage, 168
getBufferSizeBytes
uva::utils::containers::greedy_memory_storage, 168
getStorageRef
uva::utils::containers::alloc::greedy_memory_allocator, 166
gram_id_byte_len_2_type
uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id, 106
gram_id_type_2_byte_len
uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id, 106
greedy_memory_allocator
uva::utils::containers::alloc::greedy_memory_allocator, 163
greedy_memory_storage
uva::utils::containers::greedy_memory_storage, 168

h2d_map_trie
uva::smt::bpbd::server::lm::h2d_map_trie, 170
HANDLE_ENDIAN
math_utils.hpp, 411
HEADER_DELIMITER
uva::smt::bpbd::common::messaging::trans_job_request, 339
uva::smt::bpbd::common::messaging::trans_job_response, 341
handlers_map_iter_type
uva::smt::bpbd::server::trans_manager, 345
handlers_map_type
uva::smt::bpbd::server::trans_manager, 345
has_data
uva::utils::containers::dynamic_stack_array, 142
has_more
uva::utils::file::text_piece_reader, 295
has_target
uva::smt::bpbd::server::tm::models::tm_source_entry, 319
has_translations
uva::smt::bpbd::server::tm::models::tm_source_entry, 319
hashing_utils.hpp
A, 404
B, 404
C, 404
cwfold, 404
cwmixa, 405
cwmixb, 405
get16bits, 405
hashing_word_index
uva::smt::bpbd::server::lm::dictionary::hashing_word_index, 173

ID_TYPE_LEN_BYTES
uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id, 109
INFO
uva::utils::logging, 82
INFO1
uva::utils::logging, 82
INFO1_PARAM_VALUE
logger.hpp, 407
INFO2
uva::utils::logging, 82
INFO2_PARAM_VALUE
logger.hpp, 407
INFO3
uva::utils::logging, 82
INFO3_PARAM_VALUE
logger.hpp, 407
INFO_PARAM_VALUE
logger.hpp, 407
INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL
lm_gram_builder.cpp, 468
uva::smt::bpbd::server::lm::arpa, 63
INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL_WEIGHT
lm_gram_builder.cpp, 468
INSTANTIATE_LAYERED_TRIE_TEMPLATES_NAME_TYPE
layered_trie_base.hpp, 438
uva::smt::bpbd::server::lm, 55, 56
INSTANTIATE_TRIE_BUILDER_FILE_READER
lm_basic_builder.cpp, 467
uva::smt::bpbd::server::lm::arpa, 63
INSTANTIATE_TRIE_FUNCS_LEVEL
generic_trie_base.hpp, 436
INSTANTIATE_TRIE_TEMPLATE_TYPE
generic_trie_base.hpp, 436
uva::smt::bpbd::server::lm, 56, 57
IS_ENOUGH_LOGGING_LEVEL
logger.cpp, 464
IS_EQUAL
optimizing_word_index.hpp, 425
id

uva::smt::bpbd::server::lm::__C2WArrayTrie::T↩ WordIdPBData, 365
uva::smt::bpbd::server::lm::__W2CArrayTrie::S_↩ M_GramData, 263
id_manager
uva::smt::bpbd::common::messaging::id_manager, 176
inc/client/client_config.hpp, 387
inc/client/trans_job.hpp, 387
inc/client/trans_job_status.hpp, 388
inc/client/trans_manager.hpp, 389
inc/client/translation_client.hpp, 390
inc/common/messaging/id_manager.hpp, 390
inc/common/messaging/trans_job_code.hpp, 391
inc/common/messaging/trans_job_id.hpp, 391
inc/common/messaging/trans_job_request.hpp, 392
inc/common/messaging/trans_job_response.hpp, 392
inc/common/messaging/trans_session_id.hpp, 393
inc/common/utils/containers/array_utils.hpp, 393
inc/common/utils/containers/circular_queue.hpp, 395
inc/common/utils/containers/dynamic_memory_↩ arrays.hpp, 396
inc/common/utils/containers/fixed_size_hashmap.hpp, 398
inc/common/utils/containers/greedy_memory_allocator.↩ hpp, 398
inc/common/utils/containers/greedy_memory_storage.↩ hpp, 399
inc/common/utils/containers/upp_diag_matrix.hpp, 399
inc/common/utils/exceptions.hpp, 400
inc/common/utils/file/afile_reader.hpp, 401
inc/common/utils/file/cstyle_file_reader.hpp, 401
inc/common/utils/file/file_stream_reader.hpp, 402
inc/common/utils/file/memory_mapped_file_reader.hpp, 402
inc/common/utils/file/text_piece_reader.hpp, 403
inc/common/utils/hashing_utils.hpp, 404
inc/common/utils/logging/logger.hpp, 405
inc/common/utils/math_utils.hpp, 410
inc/common/utils/monitor/statistics_monitor.hpp, 411
inc/common/utils/string_utils.hpp, 412
inc/common/utils/threads.hpp, 413
inc/main.hpp, 413
inc/server/cmd_line_handler.hpp, 414
inc/server/common/models/phrase_uid.hpp, 415
inc/server/decoder/de_configs.hpp, 415
inc/server/decoder/de_configurator.hpp, 416
inc/server/decoder/de_parameters.hpp, 416
inc/server/decoder/sentence/sentence_data_map.hpp, 416
inc/server/decoder/sentence/sentence_decoder.hpp, 417
inc/server/decoder/stack/multi_stack.hpp, 418
inc/server/decoder/stack/stack_data.hpp, 418
inc/server/decoder/stack/stack_level.hpp, 419
inc/server/decoder/stack/stack_state.hpp, 419
inc/server/decoder/stack/state_data.hpp, 420
inc/server/lm/builders/lm_basic_builder.hpp, 420
inc/server/lm/builders/lm_gram_builder.hpp, 421
inc/server/lm/builders/lm_gram_builder_factory.hpp, 421
inc/server/lm/dictionaries/aword_index.hpp, 422
inc/server/lm/dictionaries/basic_word_index.hpp, 422
inc/server/lm/dictionaries/counting_word_index.hpp, 423
inc/server/lm/dictionaries/hashing_word_index.hpp, 423
inc/server/lm/dictionaries/optimizing_word_index.hpp, 424
inc/server/lm/lm_configs.hpp, 426
inc/server/lm/lm_configurator.hpp, 426
inc/server/lm/lm_consts.hpp, 427
inc/server/lm/lm_executor.hpp, 428
inc/server/lm/lm_parameters.hpp, 428
inc/server/lm/mgrams/m_gram_id.hpp, 429
inc/server/lm/mgrams/m_gram_id_tables.hpp, 430
inc/server/lm/mgrams/m_gram_payload.hpp, 430
inc/server/lm/mgrams/model_m_gram.hpp, 431
inc/server/lm/mgrams/query_m_gram.hpp, 431
inc/server/lm/models/bitmap_hash_cache.hpp, 432
inc/server/lm/models/c2d_hybrid_trie.hpp, 432
inc/server/lm/models/c2d_map_trie.hpp, 433
inc/server/lm/models/c2w_array_trie.hpp, 433
inc/server/lm/models/g2d_map_trie.hpp, 434
inc/server/lm/models/generic_trie_base.hpp, 435
inc/server/lm/models/h2d_map_trie.hpp, 437
inc/server/lm/models/layered_trie_base.hpp, 437
inc/server/lm/models/m_gram_query.hpp, 439
inc/server/lm/models/w2c_array_trie.hpp, 439
inc/server/lm/models/w2c_hybrid_trie.hpp, 440
inc/server/lm/models/w2ch_um_storage.hpp, 441
inc/server/lm/models/word_index_trie_base.hpp, 441
inc/server/lm/proxy/lm_fast_query_proxy.hpp, 442
inc/server/lm/proxy/lm_fast_query_proxy_local.hpp, 442
inc/server/lm/proxy/lm_proxy.hpp, 443
inc/server/lm/proxy/lm_proxy_local.hpp, 443
inc/server/lm/proxy/lm_slow_query_proxy.hpp, 444
inc/server/lm/proxy/lm_slow_query_proxy_local.hpp, 444
inc/server/rm/builders/rm_basic_builder.hpp, 445
inc/server/rm/models/rm_basic_model.hpp, 445
inc/server/rm/models/rm_entry.hpp, 446
inc/server/rm/models/rm_query.hpp, 446
inc/server/rm/proxy/rm_proxy.hpp, 447
inc/server/rm/proxy/rm_proxy_local.hpp, 447
inc/server/rm/proxy/rm_query_proxy.hpp, 448
inc/server/rm/proxy/rm_query_proxy_local.hpp, 448
inc/server/rm/rm_configs.hpp, 448
inc/server/rm/rm_configurator.hpp, 449
inc/server/rm/rm_consts.hpp, 449
inc/server/rm/rm_parameters.hpp, 450
inc/server/server_configs.hpp, 450
inc/server/server_consts.hpp, 451
inc/server/server_parameters.hpp, 451
inc/server/tm/builders/tm_basic_builder.hpp, 452
inc/server/tm/models/tm_basic_model.hpp, 452
inc/server/tm/models/tm_query.hpp, 453
inc/server/tm/models/tm_source_entry.hpp, 453

inc/server/tm/models/tm_target_entry.hpp, 454
inc/server/tm/proxy/tm_proxy.hpp, 454
inc/server/tm/proxy/tm_proxy_local.hpp, 455
inc/server/tm/proxy/tm_query_proxy.hpp, 455
inc/server/tm/proxy/tm_query_proxy_local.hpp, 456
inc/server/tm/tm_configs.hpp, 456
inc/server/tm/tm_configurator.hpp, 457
inc/server/tm/tm_consts.hpp, 457
inc/server/tm/tm_parameters.hpp, 457
inc/server/trans_job.hpp, 388
inc/server/trans_job_pool.hpp, 458
inc/server/trans_manager.hpp, 389
inc/server/trans_task.hpp, 459
inc/server/trans_task_id.hpp, 459
inc/server/trans_task_pool.hpp, 459
inc/server/trans_task_pool_worker.hpp, 460
inc/server/translation_server.hpp, 460
initialize_future_costs
uva::smt::bpbd::server::decoder::sentence↩
::sentence_decoder, 267
insert_as_first
uva::smt::bpbd::server::decoder::stack::stack_↩
level, 274
insert_as_last
uva::smt::bpbd::server::decoder::stack::stack_↩
level, 274
insert_before
uva::smt::bpbd::server::decoder::stack::stack_↩
level, 274
insert_between
uva::smt::bpbd::server::decoder::stack::stack_↩
level, 274
is_above_threshold
uva::smt::bpbd::server::decoder::stack::stack_↩
state_templ, 280
is_busy
uva::smt::bpbd::server::trans_task_pool_worker,
358
is_context_needed
uva::smt::bpbd::server::lm::generic_trie_base, 158
uva::smt::bpbd::server::lm::layered_trie_base, 180
is_curr_uni_gram
uva::smt::bpbd::server::lm::m_gram_query, 215
is_equal_last
uva::utils::containers::circular_queue, 128
is_equal_m_grams_id
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 106
is_good
uva::smt::bpbd::common::messaging::trans_job↩
_response, 341
is_good_features
uva::smt::bpbd::server::tm::builders::tm_basic_↩
builder, 299
is_hash_cached
uva::smt::bpbd::server::lm::caching::BitmapHash↩
Cache, 98
is_job_finished
uva::smt::bpbd::server::trans_job, 328
is_job_id_defined
uva::smt::bpbd::common::messaging::trans_job↩
_response, 341
is_less
uva::utils::containers::utils, 76
is_less_m_grams_id
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 108
is_lm_weight
uva::smt::bpbd::server::lm::lm_parameters, 196
is_m_gram_potentially_present
uva::smt::bpbd::server::lm::generic_trie_base, 158
is_more_m_grams_id
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 108
is_not_finished
uva::smt::bpbd::server::lm::m_gram_query, 215
is_num_entries_needed
uva::smt::bpbd::server::rm::models::rm_basic_↩
model, 245
uva::smt::bpbd::server::tm::models::tm_basic_↩
model, 304
is_open
uva::utils::file::afile_reader, 89
uva::utils::file::cstyle_file_reader, 136
uva::utils::file::file_stream_reader, 147
uva::utils::file::memory_mapped_file_reader, 221
is_post_actions_needed
uva::smt::bpbd::server::lm::dictionary::aword_↩
index, 91
uva::smt::bpbd::server::lm::dictionary::basic_↩
word_index, 96
uva::smt::bpbd::server::lm::dictionary::counting_↩
word_index, 133
uva::smt::bpbd::server::lm::dictionary::hashing_↩
word_index, 174
uva::smt::bpbd::server::lm::dictionary::optimizing↩
_word_index, 229
is_post_grams
uva::smt::bpbd::server::lm::c2w_array_trie, 125
uva::smt::bpbd::server::lm::w2c_array_trie, 373
uva::smt::bpbd::server::lm::word_index_trie_base,
383
is_pre_process
uva::smt::bpbd::client::client_config, 131
is_progress_bar_on
uva::utils::logging::logger, 207
is_relevant_level
uva::utils::logging::logger, 208
is_space_left
uva::smt::bpbd::server::decoder::stack::stack_↩
level, 275
is_stop_running
uva::smt::bpbd::server::trans_job_pool, 334
is_unk_entry
uva::smt::bpbd::server::rm::models::rm_basic_↩
model, 245

uva::smt::bpbd::server::tm::models::tm_basic_↩
model, 304
is_unk_trans
uva::smt::bpbd::server::tm::models::tm_target_↩
entry_temp, 323
is_unk_unigram
uva::smt::bpbd::server::lm::m_grams::model_m_↩
gram, 224
is_word_counts_needed
uva::smt::bpbd::server::lm::dictionary::aword_↩
index, 91
uva::smt::bpbd::server::lm::dictionary::basic_↩
word_index, 96
uva::smt::bpbd::server::lm::dictionary::counting_↩
word_index, 133
uva::smt::bpbd::server::lm::dictionary::hashing_↩
word_index, 174
uva::smt::bpbd::server::lm::dictionary::optimizing↩
_word_index, 230
is_word_index_continuous
uva::smt::bpbd::server::lm::dictionary::aword_↩
index, 91
uva::smt::bpbd::server::lm::dictionary::basic_↩
word_index, 96
uva::smt::bpbd::server::lm::dictionary::counting_↩
word_index, 133
uva::smt::bpbd::server::lm::dictionary::hashing_↩
word_index, 174
uva::smt::bpbd::server::lm::dictionary::optimizing↩
_word_index, 230
is_word_registering_needed
uva::smt::bpbd::server::lm::dictionary::aword_↩
index, 92
uva::smt::bpbd::server::lm::dictionary::basic_↩
word_index, 96
uva::smt::bpbd::server::lm::dictionary::counting_↩
word_index, 134
uva::smt::bpbd::server::lm::dictionary::hashing_↩
word_index, 174
uva::smt::bpbd::server::lm::dictionary::optimizing↩
_word_index, 230

job_id_type
uva::smt::bpbd::common::messaging, 39
jobs_list_iter_type
uva::smt::bpbd::client::trans_manager, 348
uva::smt::bpbd::server::trans_job_pool, 332
jobs_list_type
uva::smt::bpbd::client::trans_manager, 348
uva::smt::bpbd::server::trans_job_pool, 332
jobs_map_iter_type
uva::smt::bpbd::client::trans_manager, 348
uva::smt::bpbd::server::trans_job_pool, 332
jobs_map_type
uva::smt::bpbd::client::trans_manager, 348
uva::smt::bpbd::server::trans_job_pool, 332

LAYERED_BASE_ENSURE_CONTEXT
layered_trie_base.hpp, 438
LEVEL_2_GRAM_TO_BYTE_LEN
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 109
LEVEL_2_GRAM_TO_TYPE_LEN
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 109
LEVEL_3_GRAM_TO_BYTE_LEN
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 110
LEVEL_3_GRAM_TO_TYPE_LEN
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 110
LEVEL_4_GRAM_TO_BYTE_LEN
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 110
LEVEL_4_GRAM_TO_TYPE_LEN
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 111
LEVEL_5_GRAM_TO_BYTE_LEN
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 111
LEVEL_5_GRAM_TO_TYPE_LEN
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 111
LEVEL_6_GRAM_TO_BYTE_LEN
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 112
LEVEL_6_GRAM_TO_TYPE_LEN
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 113
LINE_STRING
logger.hpp, 407
LINEAR
uva::utils::containers, 73
LOG_10
uva::utils::containers, 73
LOG_2
uva::utils::containers, 73
LOG_DEBUG
logger.hpp, 408
LOG_DEBUG1
logger.hpp, 408
LOG_DEBUG2
logger.hpp, 408
LOG_DEBUG3
logger.hpp, 408
LOG_DEBUG4
logger.hpp, 408
LOG_ERROR
logger.hpp, 408
LOG_INFO
logger.hpp, 408
LOG_INFO1
logger.hpp, 408
LOG_INFO2
logger.hpp, 408
LOG_INFO3
logger.hpp, 408

LOG_RESULT
logger.hpp, 408
LOG_USAGE
logger.hpp, 408
LOG_WARNING
logger.hpp, 409
LOGGER
logger.hpp, 409
LOGGER_DEBUG
logger.hpp, 409
layered_trie_base
uva::smt::bpbd::server::lm::layered_trie_base, 178
layered_trie_base.hpp
INSTANTIATE_LAYERED_TRIE_TEMPLATES↩
_NAME_TYPE, 438
LAYERED_BASE_ENSURE_CONTEXT, 438
length
uva::utils::file::text_piece_reader, 295
lm_basic_builder
uva::smt::bpbd::server::lm::arpa::lm_basic_builder, 181
lm_basic_builder.cpp
INSTANTIATE_TRIE_BUILDER_FILE_READER, 467
lm_builder_type
uva::smt::bpbd::server::lm, 51
lm_fast_query_proxy_local
uva::smt::bpbd::server::lm::proxy::lm_fast_query↩
_proxy_local, 187
lm_gram_builder
uva::smt::bpbd::server::lm::arpa::lm_gram_builder, 191
lm_gram_builder.cpp
INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL, 468
INSTANTIATE_ARPA_GRAM_BUILDER_LEVE↩
L_WEIGHT, 468
lm_model_reader
uva::smt::bpbd::server::lm, 51
lm_model_type
uva::smt::bpbd::server::lm, 51
lm_proxy_local
uva::smt::bpbd::server::lm::proxy::lm_proxy_local, 199
lm_query.cpp
create_arguments_parser, 470
destroy_arguments_parser, 470
main, 470
PROGRAM_VERSION_STR, 470
lm_slow_query_proxy_local
uva::smt::bpbd::server::lm::proxy::lm_slow_↩
query_proxy_local, 203
lm_word_index
uva::smt::bpbd::server::lm, 51
load_model_data
uva::smt::bpbd::server::rm::proxy::rm_proxy_local, 255
uva::smt::bpbd::server::tm::proxy::tm_proxy_local, 311
log2
uva::utils::math::const_expr, 83
log_model_type_info
uva::smt::bpbd::server::lm::c2d_hybrid_trie, 117
uva::smt::bpbd::server::lm::c2d_map_trie, 121
uva::smt::bpbd::server::lm::c2w_array_trie, 125
uva::smt::bpbd::server::lm::g2d_map_trie, 154
uva::smt::bpbd::server::lm::generic_trie_base, 160
uva::smt::bpbd::server::lm::h2d_map_trie, 172
uva::smt::bpbd::server::lm::w2c_array_trie, 373
uva::smt::bpbd::server::lm::w2c_hybrid_trie, 377
uva::smt::bpbd::server::rm::models::rm_basic_↩
model, 245
uva::smt::bpbd::server::tm::models::tm_basic_↩
model, 304
log_reader_type_info
uva::utils::file::afile_reader, 89
uva::utils::file::cstyle_file_reader, 136
uva::utils::file::file_stream_reader, 147
uva::utils::file::memory_mapped_file_reader, 221
logger.cpp
IS_ENOUGH_LOGGING_LEVEL, 464
logger.hpp
__FILENAME__, 406
DEBUG1_PARAM_VALUE, 406
DEBUG2_PARAM_VALUE, 407
DEBUG3_PARAM_VALUE, 407
DEBUG4_PARAM_VALUE, 407
DEBUG_PARAM_VALUE, 407
END_LOG, 407
ERROR_PARAM_VALUE, 407
INFO1_PARAM_VALUE, 407
INFO2_PARAM_VALUE, 407
INFO3_PARAM_VALUE, 407
INFO_PARAM_VALUE, 407
LINE_STRING, 407
LOG_DEBUG, 408
LOG_DEBUG1, 408
LOG_DEBUG2, 408
LOG_DEBUG3, 408
LOG_DEBUG4, 408
LOG_ERROR, 408
LOG_INFO, 408
LOG_INFO1, 408
LOG_INFO2, 408
LOG_INFO3, 408
LOG_RESULT, 408
LOG_USAGE, 408
LOG_WARNING, 409
LOGGER, 409
LOGGER_DEBUG, 409
PROGRESS_UPDATE_PERIOD, 409
RESULT_PARAM_VALUE, 409
SSTR, 409
STRINGIZE, 409
STRINGIZE2, 409
USAGE_PARAM_VALUE, 409

WARNING_PARAM_VALUE, 410
WHITE_SPACE_SEPARATOR, 410

m_add_garm_func
uva::smt::bpbd::server::lm::arpa::lm_gram_builder, 193
m_add_state
uva::smt::bpbd::server::decoder::stack::stack_↩ data, 270
m_back
uva::smt::bpbd::server::lm::m_grams::m_gram_↩ payload_s, 210
m_back_off
uva::smt::bpbd::server::lm::m_grams::model_m_↩ gram, 225
m_begin_ch_idx
uva::smt::bpbd::server::decoder::sentence↩ ::phrase_data_entry, 236
m_begin_lm_level
uva::smt::bpbd::server::decoder::stack::state_↩ data_templ, 286
m_begin_tag_uid
uva::smt::bpbd::server::lm::proxy::lm_proxy_local, 201
m_condition
uva::smt::bpbd::server::trans_task_pool, 357
m_conn_string
uva::smt::bpbd::server::lm::lm_parameters, 196
uva::smt::bpbd::server::rm::rm_parameters, 252
uva::smt::bpbd::server::tm::tm_parameters, 308
m_covered
uva::smt::bpbd::server::decoder::stack::state_↩ data_templ, 286
m_curr_begin_word_idx
uva::smt::bpbd::server::lm::m_gram_query, 217
m_curr_end_word_idx
uva::smt::bpbd::server::lm::m_gram_query, 217
m_de_params
uva::smt::bpbd::server::server_parameters, 268
m_distortion
uva::smt::bpbd::server::decoder::de_parameters↩ _struct, 139
m_end_ch_idx
uva::smt::bpbd::server::decoder::sentence↩ ::phrase_data_entry, 237
m_end_tag_uid
uva::smt::bpbd::server::lm::proxy::lm_proxy_local, 201
m_ext_dist_left
uva::smt::bpbd::server::decoder::de_parameters↩ _struct, 139
m_gram_id.hpp
DECLARE_STACK_GRAM_ID, 429
MAX_N_GRAM_ID_LEN_BYTES, 429
N_GRAM_ID_TYPE_LEN_BYTES, 429
m_gram_id_tables.hpp
BYTE_M_GRAM_ID_TABLES_HPP, 430
m_gram_id_type
uva::smt::bpbd::server::lm::m_grams::phrase_↩ base, 232
m_gram_payload
uva::smt::bpbd::server::lm::m_grams, 66
m_gram_payload_s
uva::smt::bpbd::server::lm::m_grams::m_gram_↩ payload_s, 210
m_gram_query
uva::smt::bpbd::server::lm::m_gram_query, 212
m_id
uva::smt::bpbd::server::lm::__G2DMapTrie::S_M↩ _GramData, 265
uva::smt::bpbd::server::lm::__H2DMapTrie::S_M↩ _GramData, 262
uva::smt::bpbd::server::lm::m_grams::m_gram_↩ id::T_Gram_Id_Key, 288
m_is_dist
uva::smt::bpbd::server::decoder::de_parameters↩ _struct, 139
m_is_recombine
uva::smt::bpbd::server::decoder::de_parameters↩ _struct, 139
m_is_stop
uva::smt::bpbd::server::decoder::stack::stack_↩ data, 270
m_lambdas
uva::smt::bpbd::server::lm::lm_parameters, 196
uva::smt::bpbd::server::rm::rm_parameters, 252
uva::smt::bpbd::server::tm::tm_parameters, 308
m_len
optimizing_word_index.hpp, 425
uva::smt::bpbd::server::lm::dictionary::__optimizing↩ _word_index::word_index_bucket_entry, 382
m_len_bytes
uva::smt::bpbd::server::lm::m_grams::m_gram_↩ id::T_Gram_Id_Key, 288
m_lm_params
uva::smt::bpbd::server::lm::__executor::lm_exec↩ _params, 184
uva::smt::bpbd::server::server_parameters, 268
m_lm_query
uva::smt::bpbd::server::decoder::stack::stack_↩ data, 270
m_m_gram
uva::smt::bpbd::server::lm::arpa::lm_gram_builder, 193
m_max_idx
uva::utils::containers::upp_diag_matrix, 368
m_max_s_phrase_len
uva::smt::bpbd::server::decoder::de_parameters↩ _struct, 139
m_max_sent
uva::smt::bpbd::client::client_config, 131
m_max_t_phrase_len
uva::smt::bpbd::server::decoder::de_parameters↩ _struct, 139
m_mem_strat
uva::smt::bpbd::server::lm::__W2CArrayTrie::S_↩

M_GramData, 263
m_min_idx
uva::utils::containers::upp_diag_matrix, 368
m_min_sent
uva::smt::bpbd::client::client_config, 131
m_min_tran_prob
uva::smt::bpbd::server::tm::tm_parameters, 308
m_model
uva::smt::bpbd::server::lm::proxy::lm_proxy_local, 201
m_next_new_word_id
uva::smt::bpbd::server::lm::dictionary::basic_word_index, 97
m_num_best_trans
uva::smt::bpbd::server::decoder::de_parameters_struct, 139
m_num_lambdas
uva::smt::bpbd::server::lm::lm_parameters, 196
uva::smt::bpbd::server::rm::rm_parameters, 252
uva::smt::bpbd::server::tm::tm_parameters, 308
m_num_sentences
uva::smt::bpbd::client::trans_job, 325
m_num_threads
uva::smt::bpbd::server::server_parameters, 268
m_num_unk_features
uva::smt::bpbd::server::tm::tm_parameters, 308
m_p_alloc
uva::smt::bpbd::server::lm::W2CH_UM_StorageFactory, 381
m_params
uva::smt::bpbd::server::decoder::stack::stack_data, 270
uva::smt::bpbd::server::lm::arpa::lm_gram_builder, 193
m_partial_score
uva::smt::bpbd::server::decoder::stack::state_data_templ, 286
m_payload
uva::smt::bpbd::server::lm::__G2DMapTrie::S_M_GramData, 265
uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_GramData, 262
uva::smt::bpbd::server::lm::m_grams::model_m_gram, 225
m_phrase_penalty
uva::smt::bpbd::server::decoder::de_parameters_struct, 140
m_phrase_uid
uva::smt::bpbd::server::decoder::sentence::phrase_data_entry, 237
m_port
uva::smt::bpbd::client::client_config, 131
m_prob
uva::smt::bpbd::server::lm::m_grams::m_gram_payload_s, 210
uva::smt::bpbd::server::lm::m_grams::model_m_gram, 225
m_probs
uva::smt::bpbd::server::lm::m_gram_query, 217
m_pruning_threshold
uva::smt::bpbd::server::decoder::de_parameters_struct, 140
m_query_file_name
uva::smt::bpbd::server::lm::__executor::lm_exec_params, 184
m_queue_mutex
uva::smt::bpbd::server::trans_task_pool, 357
m_request
uva::smt::bpbd::client::trans_job, 325
m_response
uva::smt::bpbd::client::trans_job, 325
m_rm_params
uva::smt::bpbd::server::server_parameters, 268
m_rm_query
uva::smt::bpbd::server::decoder::stack::stack_data, 270
m_s_begin_word_idx
uva::smt::bpbd::server::decoder::stack::state_data_templ, 286
m_s_end_word_idx
uva::smt::bpbd::server::decoder::stack::state_data_templ, 286
m_sent_data
uva::smt::bpbd::server::decoder::stack::stack_data, 270
m_server
uva::smt::bpbd::client::client_config, 131
m_server_port
uva::smt::bpbd::server::server_parameters, 268
m_source_entry
uva::smt::bpbd::server::decoder::sentence::phrase_data_entry, 237
m_source_file
uva::smt::bpbd::client::client_config, 131
m_source_lang
uva::smt::bpbd::client::client_config, 131
uva::smt::bpbd::server::server_parameters, 268
m_source_sent
uva::smt::bpbd::server::decoder::stack::stack_data, 270
m_stack_capacity
uva::smt::bpbd::server::decoder::de_parameters_struct, 140
m_stack_data
uva::smt::bpbd::server::decoder::stack::state_data_templ, 286
m_stack_level
uva::smt::bpbd::server::decoder::stack::state_data_templ, 287
m_status
uva::smt::bpbd::client::trans_job, 325
m_stop
uva::smt::bpbd::server::trans_task_pool, 357
m_target
uva::smt::bpbd::server::decoder::stack::state_data_templ, 287

m_target_file
uva::smt::bpbd::client::client_config, 131
m_target_lang
uva::smt::bpbd::client::client_config, 131
uva::smt::bpbd::server::server_parameters, 268
m_tasks
uva::smt::bpbd::server::trans_task_pool, 357
m_tm_params
uva::smt::bpbd::server::server_parameters, 269
m_token
uva::smt::bpbd::server::lm::arpa::lm_gram_builder, 193
m_total_score
uva::smt::bpbd::server::decoder::stack::state_↩ data_templ, 287
m_trans_frame
uva::smt::bpbd::server::decoder::stack::state_↩ data_templ, 287
m_trans_limit
uva::smt::bpbd::server::tm::tm_parameters, 308
m_unk_features
uva::smt::bpbd::server::tm::tm_parameters, 308
m_unk_word_prob
uva::smt::bpbd::server::lm::proxy::lm_proxy_local, 201
m_word
optimizing_word_index.hpp, 425
uva::smt::bpbd::server::lm::dictionary::__optimizing↩ _word_index::word_index_bucket_entry, 382
m_word_id
optimizing_word_index.hpp, 425
uva::smt::bpbd::server::lm::dictionary::__optimizing↩ _word_index::word_index_bucket_entry, 382
m_word_idx
uva::smt::bpbd::server::lm::arpa::lm_gram_builder, 193
m_word_index
uva::smt::bpbd::server::lm::proxy::lm_proxy_local, 201
uva::smt::bpbd::server::lm::word_index_trie_base, 384
m_word_index_alloc_ptr
uva::smt::bpbd::server::lm::dictionary::basic_↩ word_index, 97
m_word_index_map_ptr
uva::smt::bpbd::server::lm::dictionary::basic_↩ word_index, 97
m_word_index_mem_factor
uva::smt::bpbd::server::lm::dictionary::basic_↩ word_index, 97
m_word_penalty
uva::smt::bpbd::server::decoder::de_parameters↩ _struct, 140
MAX_ELEMENT_INDEX
uva::utils::containers::fixed_size_hashmap, 150
MAX_ID_LEN_BYTES
uva::smt::bpbd::server::lm::m_grams::m_gram_↩ id::Byte_M_Gram_Id, 114
MAX_N_GRAM_ID_LEN_BYTES
m_gram_id.hpp, 429
MAX_NUM_TOKENS_NGRAM_STR
uva::smt::bpbd::server::lm::arpa::lm_gram_builder, 193
MAX_SIZE_TYPE_VALUE
uva::utils::containers::dynamic_stack_array, 144
MAX_STACK_TRACE_LEN
main.hpp, 414
MAX_VALUE_IN_BYTES
byte_m_gram_id.cpp, 471
MGRAM_IDX_OFFSET
uva::smt::bpbd::server::lm::generic_trie_base, 160
MGramStatusEnum
uva::smt::bpbd::server::lm, 55
MIN_ELEMENT_INDEX
uva::utils::containers::fixed_size_hashmap, 151
MIN_NUM_TOKENS_NGRAM_STR
uva::smt::bpbd::server::lm::arpa::lm_gram_builder, 193
MIN_SENTENCES_PER_REQUEST
uva::smt::bpbd::client::trans_manager, 350
MONOTONE_ORIENT
uva::smt::bpbd::server::rm::models, 69
main
bpbd_client.cpp, 461
bpbd_server.cpp, 466
lm_query.cpp, 470
main.hpp
GET_ASSERT, 414
MAX_STACK_TRACE_LEN, 414
SAFE_DESTROY, 414
math_utils.hpp
BYTE_IDX, 411
BYTES_TO_BITS, 411
HANDLE_ENDIAN, 411
NUM_BITS_REMAINDER, 411
NUM_BYTES_4_BITS, 411
NUM_FULL_BYTES, 411
REMAINING_BIT_IDX, 411
VALUE_LEN_BYTES, 411
max_size
uva::utils::containers::alloc::greedy_memory_↩ allocator, 166
mem_inc_types_enum
uva::utils::containers, 73
mem_increase_strategy
uva::utils::containers::mem_increase_strategy, 218, 219
memory_mapped_file_reader
uva::utils::file::memory_mapped_file_reader, 221
memory_usage
uva::utils::monitor::memory_usage, 222
merge_recomb_from
uva::smt::bpbd::server::decoder::stack::stack_↩ state_templ, 281
model_m_gram

uva::smt::bpbd::server::lm::m_grams::model_m_↩
gram, 224
multi_stack
uva::smt::bpbd::server::decoder::stack::multi_↩
stack, 226
mv
uva::utils::logging::logging_synch, 209
my_bsearch
uva::utils::containers::utils, 76
my_bsearch_id
uva::utils::containers::utils, 77
my_bsearch_wordId_ctxId
uva::utils::containers::utils, 78
my_isearch_id
uva::utils::containers::utils, 78
my_lsearch_id
uva::utils::containers::utils, 79
my_sort
uva::utils::containers::utils, 79, 80

N_GRAM_ID_TYPE_LEN_BYTES
m_gram_id.hpp, 429
N_GRAM_IDX_IN_M_N_ARR
uva::smt::bpbd::server::lm::generic_trie_base, 160
NEEDS_BITMAP_HASH_CACHE
uva::smt::bpbd::server::lm::generic_trie_base, 161
NEW_LINE_HEADER_ENDING
uva::smt::bpbd::common::messaging::trans_job↩
_request, 339
uva::smt::bpbd::common::messaging::trans_job↩
_response, 341
NO_ELEMENT_INDEX
uva::utils::containers::fixed_size_hashmap, 151
NULL_FUNC_PTR
uva::utils::containers::ELEMENT_DEALLOC_FU↩
NC, 145
NUM_BITS_REMAINDER
math_utils.hpp, 411
NUM_BYTES_4_BITS
math_utils.hpp, 411
NUM_BYTES_WORD_ID
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 114
NUM_FEATURES
uva::smt::bpbd::server::rm::models::rm_entry_↩
temp, 251
uva::smt::bpbd::server::tm::models::tm_target_↩
entry_temp, 324
NUM_FULL_BYTES
math_utils.hpp, 411
NUM_M_GRAM_LEVELS
uva::smt::bpbd::server::lm::generic_trie_base, 161
NUM_M_N_GRAM_LEVELS
uva::smt::bpbd::server::lm::generic_trie_base, 161
NUMBER_ID_TYPES_PER_LEVEL
uva::smt::bpbd::server::lm::m_grams::m_gram_↩
id::Byte_M_Gram_Id, 114
notify_conn_closed
uva::smt::bpbd::client::trans_manager, 349
notify_job_done
uva::smt::bpbd::server::trans_job_pool, 334
notify_job_finished
uva::smt::bpbd::server::trans_manager, 346
notify_jobs_done
uva::smt::bpbd::client::trans_manager, 349
notify_jobs_sent
uva::smt::bpbd::client::trans_manager, 349
notify_task_cancel
uva::smt::bpbd::server::trans_task_pool, 356
notify_task_done
uva::smt::bpbd::server::trans_job, 328
num_targets
uva::smt::bpbd::server::tm::models::tm_source_↩
entry, 319

on_close
uva::smt::bpbd::client::translation_client, 360
uva::smt::bpbd::server::translation_server, 362
on_fail
uva::smt::bpbd::client::translation_client, 360
uva::smt::bpbd::server::translation_server, 363
on_message
uva::smt::bpbd::client::translation_client, 361
uva::smt::bpbd::server::translation_server, 363
on_open
uva::smt::bpbd::client::translation_client, 361
uva::smt::bpbd::server::translation_server, 363
open_session
uva::smt::bpbd::server::trans_manager, 346
operator bool
uva::utils::file::afile_reader, 89
uva::utils::file::cstyle_file_reader, 136
uva::utils::file::file_stream_reader, 147
uva::utils::file::memory_mapped_file_reader, 221
operator int
uva::smt::bpbd::client::trans_job_status, 343
uva::smt::bpbd::common::messaging::trans_job↩
_code, 330
operator string
uva::smt::bpbd::client::trans_job_status, 343
uva::smt::bpbd::common::messaging::trans_job↩
_code, 330
operator!=
uva::smt::bpbd::server::decoder::stack::stack_↩
state_templ, 281
uva::utils::containers::alloc, 75
uva::utils::file::text_piece_reader, 296
operator<
uva::smt::bpbd::client::trans_job_status, 343
uva::smt::bpbd::common::messaging::trans_job↩
_code, 330
uva::smt::bpbd::server::decoder::stack::stack_↩
state_templ, 281
uva::smt::bpbd::server::lm::__C2WArrayTrie, 59
uva::smt::bpbd::server::lm::__W2CArrayTrie, 61, 62
uva::smt::bpbd::server::lm::dictionary::__counting↩
_word_index, 64

operator<<
uva::smt::bpbd::client, 38
uva::smt::bpbd::common::messaging, 40
uva::smt::bpbd::server::lm, 57
uva::smt::bpbd::server::lm::m_gram_query, 216
uva::smt::bpbd::server::lm::m_grams, 66
uva::smt::bpbd::server::lm::m_grams::model_m_gram, 225
uva::smt::bpbd::server::lm::m_grams::query_m_gram, 238
uva::smt::bpbd::server::rm::models::rm_entry_temp, 251
uva::utils::file, 81
uva::utils::logging, 82
operator>
uva::smt::bpbd::server::lm::__C2WArrayTrie, 59
operator()
uva::smt::bpbd::server::trans_task_pool_worker, 358
operator=
uva::smt::bpbd::client::trans_job_status, 344
uva::smt::bpbd::common::messaging::trans_job_code, 330
uva::smt::bpbd::server::decoder::de_parameters_struct, 139
operator==
uva::smt::bpbd::client::trans_job_status, 344
uva::smt::bpbd::common::messaging::trans_job_code, 331
uva::smt::bpbd::server::decoder::stack::stack_state_templ, 282
uva::smt::bpbd::server::lm::__C2WArrayTrie, 59
uva::smt::bpbd::server::lm::__G2DMapTrie::S_M_GramData, 265
uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_GramData, 262
uva::smt::bpbd::server::rm::models::rm_entry_temp, 250
uva::smt::bpbd::server::tm::models::tm_source_entry, 320
uva::utils::containers::alloc, 75
uva::utils::file::text_piece_reader, 296, 297
operator[]
uva::smt::bpbd::server::lm::W2CH_UM_Storage, 379
uva::smt::bpbd::server::lm::m_gram_query, 215
uva::smt::bpbd::server::lm::m_grams::phrase_base, 235
uva::smt::bpbd::server::rm::models::rm_entry_temp, 250
uva::utils::containers::dynamic_stack_array, 143
uva::utils::containers::upp_diag_matrix, 367
uva::utils::file::text_piece_reader, 297
optimizing_word_index
uva::smt::bpbd::server::lm::dictionary::optimizing_word_index, 228
optimizing_word_index.hpp
IS_EQUAL, 425
m_len, 425
m_word, 425
m_word_id, 425
word_index_bucket_entry, 425
other
uva::utils::containers::alloc::greedy_memory_allocator::rebind, 239

PARAMETERS_SIZE_BYTES
uva::utils::containers::dynamic_stack_array, 144
PROGRAM_VERSION_STR
bpbd_client.cpp, 461
bpbd_server.cpp, 465
lm_query.cpp, 470
PROGRESS_UPDATE_PERIOD
logger.hpp, 409
parse_line
uva::smt::bpbd::server::lm::arpa::lm_gram_builder, 192
parse_rm_file
uva::smt::bpbd::server::rm::builders::rm_basic_builder, 241
parse_tm_file
uva::smt::bpbd::server::tm::builders::tm_basic_builder, 299
parse_to_gram
uva::smt::bpbd::server::lm::arpa::lm_gram_builder, 192
payload
uva::smt::bpbd::server::lm::__C2WArrayTrie::T_WordIdPBData, 365
uva::smt::bpbd::server::lm::__W2CArrayTrie::S_M_GramData, 263
payload_ptr
uva::smt::bpbd::server::lm::m_gram_query, 211
perform_command_loop
uva::smt::bpbd::server, 44
perform_translation
uva::smt::bpbd::server::decoder::sentence_::sentence_decoder, 267
phrase_base
uva::smt::bpbd::server::lm::m_grams::phrase_base, 232
phrase_data_entry
uva::smt::bpbd::server::decoder::sentence_::phrase_data_entry, 236
phrase_length
uva::smt::bpbd::server, 42
phrase_uid
uva::smt::bpbd::server, 42
plan_new_job
uva::smt::bpbd::server::trans_job_pool, 334
plan_new_task
uva::smt::bpbd::server::trans_task_pool, 356
pointer
uva::utils::containers::alloc::greedy_memory_allocator, 162
post_M_N_Grams
uva::smt::bpbd::server::lm::w2c_array_trie, 373

post_grams
uva::smt::bpbd::server::lm::c2w_array_trie, 125
uva::smt::bpbd::server::lm::w2c_array_trie, 373
uva::smt::bpbd::server::lm::word_index_trie_base, 383
post_m_grams
uva::smt::bpbd::server::lm::c2w_array_trie, 125
uva::smt::bpbd::server::lm::w2c_array_trie, 373
post_n_grams
uva::smt::bpbd::server::lm::c2w_array_trie, 126
uva::smt::bpbd::server::lm::w2c_array_trie, 374
post_process_feature
uva::smt::bpbd::server::tm::builders::tm_basic_↩ builder, 300
power
uva::utils::math::const_expr, 83
pre_allocate
uva::smt::bpbd::server::lm::c2d_hybrid_trie, 118
uva::smt::bpbd::server::lm::c2d_map_trie, 121
uva::smt::bpbd::server::lm::c2w_array_trie, 126
uva::smt::bpbd::server::lm::caching::BitmapHash↩ Cache, 99
uva::smt::bpbd::server::lm::g2d_map_trie, 154
uva::smt::bpbd::server::lm::generic_trie_base, 160
uva::smt::bpbd::server::lm::h2d_map_trie, 172
uva::smt::bpbd::server::lm::layered_trie_base, 180
uva::smt::bpbd::server::lm::w2c_array_trie, 374
uva::smt::bpbd::server::lm::w2c_hybrid_trie, 378
uva::smt::bpbd::server::lm::word_index_trie_base, 384
uva::utils::containers::dynamic_stack_array, 143
prepare_for_adding
uva::smt::bpbd::server::lm::m_grams::model_m_↩ gram, 225
print_server_commands
uva::smt::bpbd::server, 45
print_the_prompt
uva::smt::bpbd::server, 45
prob
uva::smt::bpbd::server::lm::__C2WArrayTrie::T↩ CtxIdProbData, 290
uva::smt::bpbd::server::lm::dictionary::__counting↩ _word_index::TWordInfo, 366
prob_weight
uva::smt::bpbd::server, 42
process_entry_weights
uva::smt::bpbd::server::rm::builders::rm_basic_↩ builder, 241
process_features
uva::smt::bpbd::server::tm::builders::tm_basic_↩ builder, 300
process_finished_jobs
uva::smt::bpbd::server::trans_job_pool, 334
process_input_cmd
uva::smt::bpbd::server, 45
process_source_entries
uva::smt::bpbd::server::rm::builders::rm_basic_↩ builder, 241
uva::smt::bpbd::server::tm::builders::tm_basic_↩ builder, 300
process_target_entry
uva::smt::bpbd::server::tm::builders::tm_basic_↩ builder, 300
process_task_result
uva::smt::bpbd::server::trans_task, 354
prune_states
uva::smt::bpbd::server::decoder::stack::stack_↩ level, 275
push_back
uva::utils::containers::circular_queue, 128, 130

query_m_gram
uva::smt::bpbd::server::lm::m_grams::query_m_↩ gram, 238
query_map
uva::smt::bpbd::server::rm::models::rm_query, 256
uva::smt::bpbd::server::tm::models::tm_query, 312
query_reordering_model
uva::smt::bpbd::server::decoder::sentence↩ ::sentence_decoder, 267
query_translation_model
uva::smt::bpbd::server::decoder::sentence↩ ::sentence_decoder, 267

README.md, 461
REMAINING_BIT_IDX
math_utils.hpp, 411
REPORT_COLLISION_WARNING
generic_trie_base.hpp, 436
RESULT
uva::utils::logging, 82
RESULT_CANCELED
uva::smt::bpbd::common::messaging::trans_job↩ _code, 329
RESULT_CANCELED_STR
trans_job_code.cpp, 463
RESULT_ERROR
uva::smt::bpbd::common::messaging::trans_job↩ _code, 329
RESULT_ERROR_STR
trans_job_code.cpp, 463
RESULT_OK
uva::smt::bpbd::common::messaging::trans_job↩ _code, 329
RESULT_OK_STR
trans_job_code.cpp, 463
RESULT_PARAM_VALUE
logger.hpp, 409
RESULT_PARTIAL
uva::smt::bpbd::common::messaging::trans_job↩ _code, 329
RESULT_PARTIAL_STR
trans_job_code.cpp, 463
RESULT_UNDEFINED
uva::smt::bpbd::common::messaging::trans_job↩ _code, 329
RESULT_UNDEFINED_STR

trans_job_code.cpp, 464
RESULT_UNKNOWN_STR
trans_job_code.cpp, 464
rec_scoped_lock
uva::utils::logging::logging_synch, 209
recombine_from
uva::smt::bpbd::server::decoder::stack::stack_↩
state_templ, 282
recursive_guard
uva::utils::threads, 85
reference
uva::utils::containers::alloc::greedy_memory_↩
allocator, 163
register_m_gram_cache
uva::smt::bpbd::server::lm::generic_trie_base, 160
register_word
uva::smt::bpbd::server::lm::dictionary::aword_↩
index, 92
uva::smt::bpbd::server::lm::dictionary::basic_↩
word_index, 97
uva::smt::bpbd::server::lm::dictionary::counting_↩
word_index, 134
uva::smt::bpbd::server::lm::dictionary::hashing_↩
word_index, 175
uva::smt::bpbd::server::lm::dictionary::optimizing↩
_word_index, 230
remember_best_score
uva::smt::bpbd::server::decoder::stack::stack_↩
level, 275
remove_from_level
uva::smt::bpbd::server::decoder::stack::stack_↩
level, 275
reordering_orientation
uva::smt::bpbd::server::rm::models, 68
report_final_result
uva::smt::bpbd::server::lm::proxy::lm_fast_query↩
_proxy_local, 190
uva::smt::bpbd::server::lm::proxy::lm_slow_↩
query_proxy_local, 205
report_run_time_info
uva::smt::bpbd::server::trans_job_pool, 334
uva::smt::bpbd::server::trans_manager, 346
uva::smt::bpbd::server::trans_task_pool, 356
uva::smt::bpbd::server::translation_server, 363
reserve
uva::smt::bpbd::server::lm::dictionary::aword_↩
index, 92
uva::smt::bpbd::server::lm::dictionary::basic_↩
word_index, 97
uva::smt::bpbd::server::lm::dictionary::hashing_↩
word_index, 175
uva::smt::bpbd::server::lm::dictionary::optimizing↩
_word_index, 231
reserve_mem_unordered_map
uva::utils::containers::alloc, 75
reset
uva::utils::file::afile_reader, 89
uva::utils::file::cstyle_file_reader, 136
uva::utils::file::file_stream_reader, 147
response_sender
uva::smt::bpbd::server::trans_manager, 345
response_setter
uva::smt::bpbd::client::translation_client, 359
rm_basic_builder
uva::smt::bpbd::server::rm::builders::rm_basic_↩
builder, 240
rm_basic_model
uva::smt::bpbd::server::rm::models::rm_basic_↩
model, 242
rm_builder_type
uva::smt::bpbd::server::rm, 67
rm_entry
uva::smt::bpbd::server::rm::models, 68
rm_entry_data
uva::smt::bpbd::server::decoder::stack::state_↩
data_templ, 287
rm_entry_map
uva::smt::bpbd::server::rm::models::rm_basic_↩
model, 242
rm_entry_temp
uva::smt::bpbd::server::rm::models::rm_entry_↩
temp, 249
rm_model_reader
uva::smt::bpbd::server::rm, 67
rm_model_type
uva::smt::bpbd::server::rm, 68
rm_proxy_local
uva::smt::bpbd::server::rm::proxy::rm_proxy_local,
254
rm_query
uva::smt::bpbd::server::rm::models::rm_query, 256
rm_query_proxy_local
uva::smt::bpbd::server::rm::proxy::rm_query_↩
proxy_local, 259
run
uva::smt::bpbd::server::translation_server, 363

S_M_GramData
uva::smt::bpbd::server::lm::__G2DMapTrie::S_M↩
_GramData, 264
uva::smt::bpbd::server::lm::__H2DMapTrie::S_M↩
_GramData, 261
SAFE_DESTROY
main.hpp, 414
SELF
uva::smt::bpbd::server::lm::__G2DMapTrie::S_M↩
_GramData, 264
uva::smt::bpbd::server::lm::__H2DMapTrie::S_M↩
_GramData, 261
SERVER_CONFIGS_HPP
server_configs.hpp, 450
SOURCE_UNK_UID
uva::smt::bpbd::server::rm::models::rm_basic_↩
model, 247
SSTR
logger.hpp, 409
STATUS_REQ_INITIALIZED

uva::smt::bpbd::client::trans_job_status, 342
STATUS_REQ_INITIALIZED_STR
trans_job_status.cpp, 462
STATUS_REQ_SENT_FAIL
uva::smt::bpbd::client::trans_job_status, 342
STATUS_REQ_SENT_FAIL_STR
trans_job_status.cpp, 462
STATUS_REQ_SENT_GOOD
uva::smt::bpbd::client::trans_job_status, 342
STATUS_REQ_SENT_GOOD_STR
trans_job_status.cpp, 462
STATUS_RES_RECEIVED
uva::smt::bpbd::client::trans_job_status, 342
STATUS_RES_RECEIVED_STR
trans_job_status.cpp, 462
STATUS_UNDEFINED
uva::smt::bpbd::client::trans_job_status, 342
STATUS_UNDEFINED_STR
trans_job_status.cpp, 462
STATUS_UNKNOWN_STR
trans_job_status.cpp, 463
STRINGIZE
logger.hpp, 409
STRINGIZE2
logger.hpp, 409
SWAP_ORIENT
uva::smt::bpbd::server::rm::models, 69
scoped_guard
uva::utils::threads, 85
scoped_lock
uva::smt::bpbd::common::messaging::id_manager, 176
search_m_gram_ctx_id
uva::smt::bpbd::server::lm::__LayeredTrieBase, 60
send
uva::smt::bpbd::client::translation_client, 361
send_response
uva::smt::bpbd::server::translation_server, 363
send_translation_jobs
uva::smt::bpbd::client::trans_manager, 349
sentence_data_map
uva::smt::bpbd::server::decoder::sentence, 47
sentence_decoder
uva::smt::bpbd::server::decoder::sentence::sentence_decoder, 266
serialize
uva::smt::bpbd::common::messaging::trans_job_request, 338
uva::smt::bpbd::common::messaging::trans_job_response, 341
server
uva::smt::bpbd::server::translation_server, 362
server_configs.hpp
SERVER_CONFIGS_HPP, 450
session_id_type
uva::smt::bpbd::common::messaging, 39
sessions_map_iter_type
uva::smt::bpbd::server::trans_job_pool, 332
sessions_map_type
uva::smt::bpbd::server::trans_job_pool, 333
uva::smt::bpbd::server::trans_manager, 345
set
uva::utils::file::text_piece_reader, 297
set_cache_context_id
uva::smt::bpbd::server::lm::layered_trie_base, 180
set_cancel_task_notifier
uva::smt::bpbd::server::trans_task, 354
set_curr_payload
uva::smt::bpbd::server::lm::m_gram_query, 215
set_data
uva::smt::bpbd::server::lm::m_gram_query, 216
uva::smt::bpbd::server::tm::models::tm_target_entry_temp, 323
set_decoder_params
uva::smt::bpbd::server, 45
set_def_unk_word_prob
uva::smt::bpbd::server::lm::c2d_hybrid_trie, 118
uva::smt::bpbd::server::lm::c2d_map_trie, 121
uva::smt::bpbd::server::lm::c2w_array_trie, 126
uva::smt::bpbd::server::lm::g2d_map_trie, 154
uva::smt::bpbd::server::lm::h2d_map_trie, 172
uva::smt::bpbd::server::lm::w2c_array_trie, 374
uva::smt::bpbd::server::lm::w2c_hybrid_trie, 378
uva::smt::bpbd::server::lm::word_index_trie_base, 384
set_done_job_notifier
uva::smt::bpbd::server::trans_job, 328
set_entry_uid
uva::smt::bpbd::server::rm::models::rm_entry_temp, 250
set_features
uva::smt::bpbd::server::tm::models::tm_target_entry_temp, 323
set_job_response
uva::smt::bpbd::client::trans_manager, 350
set_job_result_setter
uva::smt::bpbd::server::trans_job_pool, 334
set_log_level
uva::smt::bpbd::server, 45
set_m_gram
uva::smt::bpbd::server::lm::m_grams::query_m_gram, 238
set_num_entries
uva::smt::bpbd::server::rm::models::rm_basic_model, 245
uva::smt::bpbd::server::tm::models::tm_basic_model, 304
set_num_threads
uva::smt::bpbd::server, 46
uva::smt::bpbd::server::trans_job_pool, 336
uva::smt::bpbd::server::trans_manager, 346
uva::smt::bpbd::server::trans_task_pool, 356
uva::smt::bpbd::server::translation_server, 364
set_reporting_level
uva::utils::logging::logger, 208
set_response_sender

uva::smt::bpbd::server::trans_manager, 347
set_session_id
uva::smt::bpbd::common::messaging::trans_job_request, 338
set_source_uid
uva::smt::bpbd::server::tm::models::tm_source_entry, 320
set_tokens_and_word_ids
uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local, 206
set_unk_entry
uva::smt::bpbd::server::tm::models::tm_basic_model, 304
set_word_ids
uva::smt::bpbd::server::lm::m_grams::phrase_base, 235
set_word_indxes
uva::smt::bpbd::server::lm::m_gram_query, 216
shrink
uva::utils::containers::dynamic_stack_array, 143
size
uva::smt::bpbd::client::trans_job_status, 342
uva::smt::bpbd::common::messaging::trans_job_code, 329
uva::smt::bpbd::server::rm::models, 69
uva::utils::containers, 73
uva::utils::containers::dynamic_stack_array, 143
uva::utils::logging, 82
size_type
uva::utils::containers::alloc::greedy_memory_allocator, 163
uva::utils::containers::greedy_memory_storage, 167
sizes_map
uva::smt::bpbd::server::tm::builders, 70
sort
uva::utils::containers::dynamic_stack_array, 144
src/client/bpbd_client.cpp, 461
src/client/trans_job_status.cpp, 462
src/common/messaging/trans_job_code.cpp, 463
src/common/utils/logging/logger.cpp, 464
src/common/utils/monitor/statistics_monitor.cpp, 465
src/server/bpbd_server.cpp, 465
src/server/decoder/de_configurator.cpp, 466
src/server/lm/builders/lm_basic_builder.cpp, 466
src/server/lm/builders/lm_gram_builder.cpp, 468
src/server/lm/lm_configurator.cpp, 469
src/server/lm/lm_query.cpp, 469
src/server/lm/mgrams/byte_m_gram_id.cpp, 470
src/server/lm/mgrams/model_m_gram.cpp, 471
src/server/lm/mgrams/query_m_gram.cpp, 471
src/server/lm/models/c2d_hybrid_trie.cpp, 472
src/server/lm/models/c2d_map_trie.cpp, 472
src/server/lm/models/c2w_array_trie.cpp, 473
src/server/lm/models/g2d_map_trie.cpp, 473
src/server/lm/models/h2d_map_trie.cpp, 474
src/server/lm/models/m_gram_query.cpp, 475
src/server/lm/models/w2c_array_trie.cpp, 475
src/server/lm/models/w2c_hybrid_trie.cpp, 476
src/server/rm/rm_configurator.cpp, 476
src/server/tm/models/tm_target_entry.cpp, 477
src/server/tm/tm_configurator.cpp, 477
src/server/trans_task_pool.cpp, 477
src/server/trans_task_pool_worker.cpp, 478
stack_data
uva::smt::bpbd::server::decoder::stack::stack_data, 269
stack_level
uva::smt::bpbd::server::decoder::stack::stack_level, 271
uva::smt::bpbd::server::decoder::stack::stack_state_templ, 282
stack_level_ptr
uva::smt::bpbd::server::decoder::stack, 47
stack_state
uva::smt::bpbd::server::decoder::stack, 48
stack_state_ptr
uva::smt::bpbd::server::decoder::stack, 48
stack_state_templ
uva::smt::bpbd::server::decoder::stack::stack_state_templ, 277
start
uva::smt::bpbd::client::trans_manager, 350
start_new_m_gram
uva::smt::bpbd::server::lm::m_grams::model_m_gram, 225
start_progress_bar
uva::utils::logging::logger, 208
state_data
uva::smt::bpbd::server::decoder::stack::stack_state_templ, 276
state_data_templ
uva::smt::bpbd::server::decoder::stack::state_data_templ, 285
state_frame
uva::smt::bpbd::server::decoder::stack::state_data_templ, 284
stop
uva::smt::bpbd::client::trans_manager, 350
uva::smt::bpbd::server, 46
uva::smt::bpbd::server::trans_job_pool, 336
uva::smt::bpbd::server::trans_manager, 347
uva::smt::bpbd::server::trans_task_pool_worker, 358
uva::smt::bpbd::server::translation_server, 364
stop_progress_bar
uva::utils::logging::logger, 208
str
uva::smt::bpbd::client::trans_job_status, 344
uva::smt::bpbd::common::messaging::trans_job_code, 331
uva::utils::file::text_piece_reader, 297
string_utils.hpp
valid_digit, 413

T_M_Gram_PB_Entry
uva::smt::bpbd::server::lm::g2d_map_trie, 152

uva::smt::bpbd::server::lm::h2d_map_trie, 170
T_M_Gram_Prob_Entry
uva::smt::bpbd::server::lm::g2d_map_trie, 152
uva::smt::bpbd::server::lm::h2d_map_trie, 170
T_M_GramData
uva::smt::bpbd::server::lm::__W2CArrayTrie, 61
T_M_GramWordEntry
uva::smt::bpbd::server::lm::w2c_array_trie, 371
T_N_GramData
uva::smt::bpbd::server::lm::__W2CArrayTrie, 61
T_N_GramWordEntry
uva::smt::bpbd::server::lm::w2c_array_trie, 371
TARGET_UNK_UID
uva::smt::bpbd::server::rm::models::rm_basic_↩
model, 247
TC2DHybridTrieBasic
uva::smt::bpbd::server::lm, 51
TC2DHybridTrieCount
uva::smt::bpbd::server::lm, 51
TC2DHybridTrieHashing
uva::smt::bpbd::server::lm, 51
TC2DHybridTrieOptBasic
uva::smt::bpbd::server::lm, 51
TC2DHybridTrieOptCount
uva::smt::bpbd::server::lm, 51
TC2DMapTrieBasic
uva::smt::bpbd::server::lm, 51
TC2DMapTrieCount
uva::smt::bpbd::server::lm, 51
TC2DMapTrieHashing
uva::smt::bpbd::server::lm, 51
TC2DMapTrieOptBasic
uva::smt::bpbd::server::lm, 52
TC2DMapTrieOptCount
uva::smt::bpbd::server::lm, 52
TC2WArrayTrieBasic
uva::smt::bpbd::server::lm, 52
TC2WArrayTrieCount
uva::smt::bpbd::server::lm, 52
TC2WArrayTrieHashing
uva::smt::bpbd::server::lm, 52
TC2WArrayTrieOptBasic
uva::smt::bpbd::server::lm, 52
TC2WArrayTrieOptCount
uva::smt::bpbd::server::lm, 52
TCapacityIncFunct
uva::utils::containers, 72
TCtxIdProbEntry
uva::smt::bpbd::server::lm::c2w_array_trie, 123
TEXT_SENTENCE_DELIMITER
uva::smt::bpbd::common::messaging::trans_job↩
_request, 339
TElemType
uva::utils::containers::dynamic_stack_array, 141
uva::utils::containers::fixed_size_hashmap, 148
TG2DMapTrieBasic
uva::smt::bpbd::server::lm, 52
TG2DMapTrieCount
uva::smt::bpbd::server::lm, 52
TG2DMapTrieHashing
uva::smt::bpbd::server::lm, 52
TG2DMapTrieOptBasic
uva::smt::bpbd::server::lm, 52
TG2DMapTrieOptCount
uva::smt::bpbd::server::lm, 53
TH2DMapTrieBasic
uva::smt::bpbd::server::lm, 53
TH2DMapTrieCount
uva::smt::bpbd::server::lm, 53
TH2DMapTrieHashing
uva::smt::bpbd::server::lm, 53
TH2DMapTrieOptBasic
uva::smt::bpbd::server::lm, 53
TH2DMapTrieOptCount
uva::smt::bpbd::server::lm, 53
THROW_EXCEPTION
exceptions.hpp, 400
THROW_MUST_NOT_CALL
exceptions.hpp, 401
THROW_MUST_OVERRIDE
exceptions.hpp, 401
THROW_NOT_IMPLEMENTED
exceptions.hpp, 401
TIndexType
uva::utils::containers::dynamic_stack_array, 141
TLongId
uva::smt::bpbd::server::lm::identifiers, 65
TM_Gram_Id
uva::smt::bpbd::server::lm::__G2DMapTrie::S_M↩
_GramData, 264
uva::smt::bpbd::server::lm::__H2DMapTrie::S_M↩
_GramData, 261
TM_Gram_Id_Value_Ptr
uva::smt::bpbd::server::lm::m_grams::m_gram_id,
67
TMemotyUsage
uva::utils::monitor, 84
TRANS_JOB_POOL_HPP
trans_job_pool.hpp, 458
TShortId
uva::smt::bpbd::server::lm::identifiers, 65
TStorageData
uva::utils::containers::greedy_memory_storage,
167
TStorageMap
uva::smt::bpbd::server::lm, 53
TStorageMapAllocator
uva::smt::bpbd::server::lm, 53
TStorageMapEntry
uva::smt::bpbd::server::lm, 53
TStorageUnsignedMap
uva::smt::bpbd::server::lm, 53
TW2CArrayTrieBasic
uva::smt::bpbd::server::lm, 53
TW2CArrayTrieCount
uva::smt::bpbd::server::lm, 54

TW2CArrayTrieHashing
uva::smt::bpbd::server::lm, 54
TW2CArrayTrieOptBasic
uva::smt::bpbd::server::lm, 54
TW2CArrayTrieOptCount
uva::smt::bpbd::server::lm, 54
TW2CHybridTrieBasic
uva::smt::bpbd::server::lm, 54
TW2CHybridTrieCount
uva::smt::bpbd::server::lm, 54
TW2CHybridTrieHashing
uva::smt::bpbd::server::lm, 54
TW2CHybridTrieOptBasic
uva::smt::bpbd::server::lm, 54
TW2CHybridTrieOptCount
uva::smt::bpbd::server::lm, 54
TWordIdPBEntry
uva::smt::bpbd::server::lm::c2w_array_trie, 123
TWordIndexAllocator
uva::smt::bpbd::server::lm::dictionary::basic_↩
word_index, 94
TWordIndexEntry
uva::smt::bpbd::server::lm::dictionary::basic_↩
word_index, 94
TWordIndexMap
uva::smt::bpbd::server::lm::dictionary::basic_↩
word_index, 94
TWordIndexMapConstIter
uva::smt::bpbd::server::lm::dictionary::basic_↩
word_index, 94
tail_to_string
uva::utils::containers::circular_queue, 130
task_id_type
uva::smt::bpbd::server, 42
tasks_const_iter_type
uva::smt::bpbd::server::trans_job, 326
tasks_iter_type
uva::smt::bpbd::server::trans_job, 326
tasks_list_type
uva::smt::bpbd::server::trans_job, 326
tasks_queue_iter_type
uva::smt::bpbd::server::trans_task_pool, 355
tasks_queue_type
uva::smt::bpbd::server::trans_task_pool, 355
text_piece_reader
uva::utils::file::text_piece_reader, 291
threads_list_type
uva::smt::bpbd::server::trans_task_pool, 355
tm_basic_builder
uva::smt::bpbd::server::tm::builders::tm_basic_↩
builder, 298
tm_basic_model
uva::smt::bpbd::server::tm::models::tm_basic_↩
model, 302
tm_builder_type
uva::smt::bpbd::server::tm, 69
tm_const_source_entry
uva::smt::bpbd::server::tm::models, 71
tm_const_source_entry_ptr
uva::smt::bpbd::server::tm::models, 71
tm_const_target_entry
uva::smt::bpbd::server::tm::models, 71
tm_model_reader
uva::smt::bpbd::server::tm, 69
tm_model_type
uva::smt::bpbd::server::tm, 70
tm_proxy_local
uva::smt::bpbd::server::tm::proxy::tm_proxy_local,
311
tm_query
uva::smt::bpbd::server::tm::models::tm_query, 313
tm_query_proxy_local
uva::smt::bpbd::server::tm::proxy::tm_query_↩
proxy_local, 316
tm_source_entry
uva::smt::bpbd::server::tm::models::tm_source_↩
entry, 317
tm_source_entry_map
uva::smt::bpbd::server::tm::models::tm_basic_↩
model, 302
tm_source_entry_ptr
uva::smt::bpbd::server::tm::models, 71
tm_target_entry
uva::smt::bpbd::server::tm::models, 71
tm_target_entry_temp
uva::smt::bpbd::server::tm::models::tm_target_↩
entry_temp, 321
tokens_to_string
uva::utils::file, 81
trans_job
uva::smt::bpbd::client::trans_job, 325
uva::smt::bpbd::server::trans_job, 326
trans_job_code
uva::smt::bpbd::common::messaging::trans_job↩
_code, 329, 330
trans_job_code.cpp
RESULT_CANCELED_STR, 463
RESULT_ERROR_STR, 463
RESULT_OK_STR, 463
RESULT_PARTIAL_STR, 463
RESULT_UNDEFINED_STR, 464
RESULT_UNKNOWN_STR, 464
trans_job_pool
uva::smt::bpbd::server::trans_job_pool, 333
trans_job_pool.hpp
TRANS_JOB_POOL_HPP, 458
trans_job_ptr
uva::smt::bpbd::client, 38
uva::smt::bpbd::server, 42
trans_job_request
uva::smt::bpbd::common::messaging::trans_job↩
_request, 337
trans_job_request_ptr
uva::smt::bpbd::common::messaging, 39
trans_job_response

uva::smt::bpbd::common::messaging::trans_job↩
_response, 340
trans_job_response_ptr
uva::smt::bpbd::common::messaging, 39
trans_job_status
uva::smt::bpbd::client::trans_job_status, 343
trans_job_status.cpp
STATUS_REQ_INITIALIZED_STR, 462
STATUS_REQ_SENT_FAIL_STR, 462
STATUS_REQ_SENT_GOOD_STR, 462
STATUS_RES_RECEIVED_STR, 462
STATUS_UNDEFINED_STR, 462
STATUS_UNKNOWN_STR, 463
trans_manager
uva::smt::bpbd::client::trans_manager, 349
uva::smt::bpbd::server::trans_manager, 345
trans_task
uva::smt::bpbd::server::trans_task, 352
trans_task_pool
uva::smt::bpbd::server::trans_task_pool, 355
trans_task_pool_worker
uva::smt::bpbd::server::trans_task_pool, 357
uva::smt::bpbd::server::trans_task_pool_worker,
358
trans_task_ptr
uva::smt::bpbd::server, 42
translate
uva::smt::bpbd::server::decoder::sentence↩
::sentence_decoder, 267
uva::smt::bpbd::server::trans_manager, 347
uva::smt::bpbd::server::trans_task, 354
translation_client
uva::smt::bpbd::client::translation_client, 359
translation_client.hpp
ASIO_STANDALONE, 390
translation_server
uva::smt::bpbd::server::translation_server, 362
translation_server.hpp
ASIO_STANDALONE, 461

UNDEFINED
uva::utils::containers, 73
UNDEFINED_ARR_IDX
uva::smt::bpbd::server::lm::generic_trie_base, 161
UNDEFINED_MGS
uva::smt::bpbd::server::lm, 55
UNDEFINED_WORD_IDX
uva::smt::bpbd::server::decoder::stack::state_↩
data_templ, 287
UNKNOWN_ORIENT
uva::smt::bpbd::server::rm::models, 69
UNKNOWN_TARGET_ENTRY_UID
uva::smt::bpbd::server::tm::models::tm_target_↩
entry_temp, 324
USAGE
uva::utils::logging, 82
USAGE_PARAM_VALUE
logger.hpp, 409
UTF8_ASCII_PUNCTUATIONS
uva::utils::text, 84
UTF8_ASCII_WHITESPACES
uva::utils::text, 84
UTF8_EMPTY_STRING
uva::utils::text, 84
UTF8_NEW_LINE_STRING
uva::utils::text, 84
UTF8_SPACE_STRING
uva::utils::text, 84
unigram_to_prob
uva::smt::bpbd::server::lm::arpa::lm_gram_builder,
192
unique_guard
uva::utils::threads, 85
update_progress_bar
uva::utils::logging::logger, 208
upp_diag_matrix
uva::utils::containers::upp_diag_matrix, 367
uva, 37
uva::smt, 37
uva::smt::bpbd, 37
uva::smt::bpbd::client, 37
operator<<, 38
trans_job_ptr, 38
uva::smt::bpbd::client::client_config, 130
is_pre_process, 131
m_max_sent, 131
m_min_sent, 131
m_port, 131
m_server, 131
m_source_file, 131
m_source_lang, 131
m_target_file, 131
m_target_lang, 131
uva::smt::bpbd::client::trans_job, 324
∼trans_job, 325
m_num_sentences, 325
m_request, 325
m_response, 325
m_status, 325
trans_job, 325
uva::smt::bpbd::client::trans_job_status, 342
operator int, 343
operator string, 343
operator<, 343
operator=, 344
operator==, 344
STATUS_REQ_INITIALIZED, 342
STATUS_REQ_SENT_FAIL, 342
STATUS_REQ_SENT_GOOD, 342
STATUS_RES_RECEIVED, 342
STATUS_UNDEFINED, 342
size, 342
str, 344
trans_job_status, 343
values, 342
uva::smt::bpbd::client::trans_manager, 347
∼trans_manager, 349

check_jobs_done_and_notify, 349
get_num_of_sentences, 349
jobs_list_iter_type, 348
jobs_list_type, 348
jobs_map_iter_type, 348
jobs_map_type, 348
MIN_SENTENCES_PER_REQUEST, 350
notify_conn_closed, 349
notify_jobs_done, 349
notify_jobs_sent, 349
send_translation_jobs, 349
set_job_response, 350
start, 350
stop, 350
trans_manager, 349
wait, 350
write_received_job_result, 350
write_result_to_file, 350
uva::smt::bpbd::client::translation_client, 358
~translation_client, 359
client, 359
conn_close_notifier, 359
connect, 360
disconnect, 360
get_uri, 360
on_close, 360
on_fail, 360
on_message, 361
on_open, 361
response_setter, 359
send, 361
translation_client, 359
wait_connect, 361
uva::smt::bpbd::common, 38
get_float, 38
get_integer, 38
get_string, 38
uva::smt::bpbd::common::messaging, 39
job_id_type, 39
operator<<, 40
session_id_type, 39
trans_job_request_ptr, 39
trans_job_response_ptr, 39
uva::smt::bpbd::common::messaging::id_manager
get_min_id, 176
get_next_id, 176
id_manager, 176
scoped_lock, 176
uva::smt::bpbd::common::messaging::id_manager< id_type >, 175
uva::smt::bpbd::common::messaging::job_id, 41
uva::smt::bpbd::common::messaging::session_id, 41
uva::smt::bpbd::common::messaging::trans_job_code, 329
operator int, 330
operator string, 330
operator<, 330
operator=, 330
operator==, 331
RESULT_CANCELED, 329
RESULT_ERROR, 329
RESULT_OK, 329
RESULT_PARTIAL, 329
RESULT_UNDEFINED, 329
size, 329
str, 331
trans_job_code, 329, 330
val, 331
values, 329
uva::smt::bpbd::common::messaging::trans_job_request, 336
de_serialize, 337
get_job_id, 337
get_session_id, 337
get_source_lang, 338
get_target_lang, 338
get_text, 338
HEADER_DELIMITER, 339
NEW_LINE_HEADER_ENDING, 339
serialize, 338
set_session_id, 338
TEXT_SENTENCE_DELIMITER, 339
trans_job_request, 337
uva::smt::bpbd::common::messaging::trans_job_response, 339
de_serialize, 340
get_code, 340
get_job_id, 340
get_text, 341
HEADER_DELIMITER, 341
is_good, 341
is_job_id_defined, 341
NEW_LINE_HEADER_ENDING, 341
serialize, 341
trans_job_response, 340
uva::smt::bpbd::server, 41
begins_with, 43
get_float_value, 44
get_int_value, 44
get_string_value, 44
perform_command_loop, 44
phrase_length, 42
phrase_uid, 42
print_server_commands, 45
print_the_prompt, 45
prob_weight, 42
process_input_cmd, 45
set_decoder_params, 45
set_log_level, 45
set_num_threads, 46
stop, 46
task_id_type, 42
trans_job_ptr, 42
trans_task_ptr, 42
word_uid, 42
uva::smt::bpbd::server::common, 46

uva::smt::bpbd::server::common::models, 46
uva::smt::bpbd::server::decoder, 46
de_parameters, 47
uva::smt::bpbd::server::decoder::de_configurator, 136
allocate_decoder, 137
connect, 137
disconnect, 137
dispose_decoder, 137
uva::smt::bpbd::server::decoder::de_parameters_struct, 138
de_parameters_struct, 138
finalize, 139
m_distortion, 139
m_ext_dist_left, 139
m_is_dist, 139
m_is_recombine, 139
m_max_s_phrase_len, 139
m_max_t_phrase_len, 139
m_num_best_trans, 139
m_phrase_penalty, 140
m_pruning_threshold, 140
m_stack_capacity, 140
m_word_penalty, 140
operator=, 139
uva::smt::bpbd::server::decoder::sentence, 47
sentence_data_map, 47
uva::smt::bpbd::server::decoder::sentence::phrase_data_entry, 236
~phrase_data_entry, 236
future_cost, 236
m_begin_ch_idx, 236
m_end_ch_idx, 237
m_phrase_uid, 237
m_source_entry, 237
phrase_data_entry, 236
uva::smt::bpbd::server::decoder::sentence::sentence_decoder, 265
~sentence_decoder, 266
compute_futue_costs, 266
count_words, 266
initialize_future_costs, 267
perform_translation, 267
query_reordering_model, 267
query_translation_model, 267
sentence_decoder, 266
translate, 267
uva::smt::bpbd::server::decoder::stack, 47
add_new_state_function, 47
stack_level_ptr, 47
stack_state, 48
stack_state_ptr, 48
uva::smt::bpbd::server::decoder::stack::multi_stack, 226
~multi_stack, 226
add_stack_state, 227
expand, 227
get_best_trans, 227
multi_stack, 226
uva::smt::bpbd::server::decoder::stack::stack_data, 269
m_add_state, 270
m_is_stop, 270
m_lm_query, 270
m_params, 270
m_rm_query, 270
m_sent_data, 270
m_source_sent, 270
stack_data, 269
uva::smt::bpbd::server::decoder::stack::stack_level, 270
~stack_level, 271
add_before, 271
add_last, 273
add_state, 273
expand, 273
find_recombine, 273
get_best_trans, 273
get_size, 274
insert_as_first, 274
insert_as_last, 274
insert_before, 274
insert_between, 274
is_space_left, 275
prune_states, 275
remember_best_score, 275
remove_from_level, 275
stack_level, 271
uva::smt::bpbd::server::decoder::stack::stack_state_templ
~stack_state_templ, 277
count_and_prune, 277
cut_the_tail, 279
expand, 279
expand_left, 279
expand_length, 279
expand_length_if_not_covered, 279
expand_right, 280
expand_trans, 280
get_stack_level, 280
get_translation, 280
is_above_threshold, 280
merge_recomb_from, 281
operator!=, 281
operator<, 281
operator==, 282
recombine_from, 282
stack_level, 282
stack_state_templ, 277
state_data, 276
uva::smt::bpbd::server::decoder::stack::stack_state_templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRAM_QUERY_LENGTH >, 275
uva::smt::bpbd::server::decoder::stack::state_data_templ
covered_info, 284
covered_to_string, 286
m_begin_lm_level, 286
m_covered, 286

m_partial_score, 286
m_s_begin_word_idx, 286
m_s_end_word_idx, 286
m_stack_data, 286
m_stack_level, 287
m_target, 287
m_total_score, 287
m_trans_frame, 287
rm_entry_data, 287
state_data_templ, 285
state_frame, 284
UNDEFINED_WORD_IDX, 287
ZERRO_WORD_IDX, 287
uva::smt::bpbd::server::decoder::stack::state_data_↩templ< NUM_WORDS_PER_SENTENCE, MAX_HISTORY_LENGTH, MAX_M_GRA↩M_QUERY_LENGTH >, 283
uva::smt::bpbd::server::lm, 48
__attribute__, 57
BAD_END_WORD_UNKNOWN_MGS, 55
BAD_NO_PAYLOAD_MGS, 55
DEF_UNK_WORD_LOG_PROB_WEIGHT, 57
GOOD_PRESENT_MGS, 55
INSTANTIATE_LAYERED_TRIE_TEMPLATES↩_NAME_TYPE, 55, 56
INSTANTIATE_TRIE_TEMPLATE_TYPE, 56, 57
lm_builder_type, 51
lm_model_reader, 51
lm_model_type, 51
lm_word_index, 51
MGramStatusEnum, 55
operator<<, 57
TC2DHybridTrieBasic, 51
TC2DHybridTrieCount, 51
TC2DHybridTrieHashing, 51
TC2DHybridTrieOptBasic, 51
TC2DHybridTrieOptCount, 51
TC2DMapTrieBasic, 51
TC2DMapTrieCount, 51
TC2DMapTrieHashing, 51
TC2DMapTrieOptBasic, 52
TC2DMapTrieOptCount, 52
TC2WArrayTrieBasic, 52
TC2WArrayTrieCount, 52
TC2WArrayTrieHashing, 52
TC2WArrayTrieOptBasic, 52
TC2WArrayTrieOptCount, 52
TG2DMapTrieBasic, 52
TG2DMapTrieCount, 52
TG2DMapTrieHashing, 52
TG2DMapTrieOptBasic, 52
TG2DMapTrieOptCount, 53
TH2DMapTrieBasic, 53
TH2DMapTrieCount, 53
TH2DMapTrieHashing, 53
TH2DMapTrieOptBasic, 53
TH2DMapTrieOptCount, 53
TStorageMap, 53
TStorageMapAllocator, 53
TStorageMapEntry, 53
TStorageUnsignedMap, 53
TW2CArrayTrieBasic, 53
TW2CArrayTrieCount, 54
TW2CArrayTrieHashing, 54
TW2CArrayTrieOptBasic, 54
TW2CArrayTrieOptCount, 54
TW2CHybridTrieBasic, 54
TW2CHybridTrieCount, 54
TW2CHybridTrieHashing, 54
TW2CHybridTrieOptBasic, 54
TW2CHybridTrieOptCount, 54
UNDEFINED_MGS, 55
uva::smt::bpbd::server::lm::__C2DHybridTrie, 57
uva::smt::bpbd::server::lm::__C2DMapTrie, 57
uva::smt::bpbd::server::lm::__C2WArrayTrie, 57
compare, 58
operator<, 59
operator>, 59
operator==, 59
uva::smt::bpbd::server::lm::__C2WArrayTrie::TCtxId↩ProbData, 289
ctx_id, 290
prob, 290
word_id, 290
uva::smt::bpbd::server::lm::__C2WArrayTrie::TWordId↩PBData, 365
id, 365
payload, 365
uva::smt::bpbd::server::lm::__G2DMapTrie, 59
uva::smt::bpbd::server::lm::__G2DMapTrie::S_M_↩GramData
~S_M_GramData, 264
m_id, 265
m_payload, 265
operator==, 265
S_M_GramData, 264
SELF, 264
TM_Gram_Id, 264
uva::smt::bpbd::server::lm::__G2DMapTrie::S_M_↩GramData< TPayloadType, TWordIdType >, 263
uva::smt::bpbd::server::lm::__H2DMapTrie, 60
uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_↩GramData
~S_M_GramData, 262
m_id, 262
m_payload, 262
operator==, 262
S_M_GramData, 261
SELF, 261
TM_Gram_Id, 261
uva::smt::bpbd::server::lm::__H2DMapTrie::S_M_↩GramData< TPayloadType >, 261
uva::smt::bpbd::server::lm::__LayeredTrieBase, 60
get_context_id, 60
search_m_gram_ctx_id, 60

uva::smt::bpbd::server::lm::__W2CArrayTrie, 61
operator<, 61, 62
T_M_GramData, 61
T_N_GramData, 61
uva::smt::bpbd::server::lm::__W2CArrayTrie::S_M_↩
GramData
id, 263
m_mem_strat, 263
payload, 263
uva::smt::bpbd::server::lm::__W2CArrayTrie::S_M_↩
GramData< PAYLOAD_TYPE >, 262
uva::smt::bpbd::server::lm::__W2CHybridTrie, 62
uva::smt::bpbd::server::lm::__executor, 59
uva::smt::bpbd::server::lm::__executor::lm_exec_↩
params, 183
m_lm_params, 184
m_query_file_name, 184
uva::smt::bpbd::server::lm::W2CH_UM_Storage, 378
~W2CH_UM_Storage, 379
at, 379
const_iterator, 379
end, 379
find, 379
operator[], 379
W2CH_UM_Storage, 379
uva::smt::bpbd::server::lm::W2CH_UM_StorageFactory
~W2CH_UM_StorageFactory, 380
create, 380
m_p_alloc, 381
W2CH_UM_StorageFactory, 380
uva::smt::bpbd::server::lm::W2CH_UM_Storage↩
Factory< N >, 379
uva::smt::bpbd::server::lm::arpa, 62
INSTANTIATE_ARPA_GRAM_BUILDER_LEVEL,
63
INSTANTIATE_TRIE_BUILDER_FILE_READER,
63
uva::smt::bpbd::server::lm::arpa::TAddGramFunct
func, 289
uva::smt::bpbd::server::lm::arpa::TAddGramFunct<
WordIndexType >, 289
uva::smt::bpbd::server::lm::arpa::lm_basic_builder
~lm_basic_builder, 181
build, 181
lm_basic_builder, 181
WordIndexType, 181
uva::smt::bpbd::server::lm::arpa::lm_basic_builder<
trie_type, reader_type >, 180
uva::smt::bpbd::server::lm::arpa::lm_gram_builder
~lm_gram_builder, 191
lm_gram_builder, 191
m_add_garm_func, 193
m_m_gram, 193
m_params, 193
m_token, 193
m_word_idx, 193
MAX_NUM_TOKENS_NGRAM_STR, 193
MIN_NUM_TOKENS_NGRAM_STR, 193
parse_line, 192
parse_to_gram, 192
unigram_to_prob, 192
uva::smt::bpbd::server::lm::arpa::lm_gram_builder<
WordIndexType, CURR_LEVEL, is_mult_↩
weight >, 190
uva::smt::bpbd::server::lm::arpa::lm_gram_builder_↩
factory
~lm_gram_builder_factory, 194
get_builder, 194
WordIndexType, 194
uva::smt::bpbd::server::lm::arpa::lm_gram_builder_↩
factory< TrieType >, 194
uva::smt::bpbd::server::lm::c2d_hybrid_trie
~c2d_hybrid_trie, 116
add_m_gram, 116
BASE, 116
c2d_hybrid_trie, 116
get_ctx_id, 117
get_m_gram_payload, 117
get_n_gram_payload, 117
get_unigram_payload, 117
get_unk_word_prob, 117
log_model_type_info, 117
pre_allocate, 118
set_def_unk_word_prob, 118
uva::smt::bpbd::server::lm::c2d_hybrid_trie< Word↩
IndexType >, 115
uva::smt::bpbd::server::lm::c2d_map_trie
~c2d_map_trie, 120
add_m_gram, 120
BASE, 119
c2d_map_trie, 119
get_ctx_id, 120
get_m_gram_payload, 120
get_n_gram_payload, 121
get_unigram_payload, 121
get_unk_word_prob, 121
log_model_type_info, 121
pre_allocate, 121
set_def_unk_word_prob, 121
uva::smt::bpbd::server::lm::c2d_map_trie< Word↩
IndexType >, 118
uva::smt::bpbd::server::lm::c2w_array_trie
~c2w_array_trie, 124
add_m_gram, 124
BASE, 123
c2w_array_trie, 123
get_ctx_id, 124
get_m_gram_payload, 124
get_n_gram_payload, 124
get_unigram_payload, 125
get_unk_word_prob, 125
is_post_grams, 125
log_model_type_info, 125
post_grams, 125
post_m_grams, 125
post_n_grams, 126

pre_allocate, 126
set_def_unk_word_prob, 126
TCtxIdProbEntry, 123
TWordIdPBEntry, 123
uva::smt::bpbd::server::lm::c2w_array_trie< Word↩ IndexType >, 122
uva::smt::bpbd::server::lm::c2w_array_trie< Word↩ IndexType >::TSubArrReference, 364
uva::smt::bpbd::server::lm::c2w_array_trie::TSubArr↩ Reference
begin_idx, 364
end_idx, 365
uva::smt::bpbd::server::lm::caching, 63
uva::smt::bpbd::server::lm::caching::BitmapHashCache, 98
~BitmapHashCache, 98
BitmapHashCache, 98
cache_m_gram_hash, 98
is_hash_cached, 98
pre_allocate, 99
uva::smt::bpbd::server::lm::dictionary, 63
basic_optimizing_word_index, 64
counting_optimizing_word_index, 64
uva::smt::bpbd::server::lm::dictionary::__AWordIndex, 64
uva::smt::bpbd::server::lm::dictionary::__counting_↩ word_index, 64
operator<, 64
uva::smt::bpbd::server::lm::dictionary::__counting_↩ word_index::TWordInfo, 365
prob, 366
word, 366
uva::smt::bpbd::server::lm::dictionary::__optimizing_↩ word_index, 64
__attribute__, 65
uva::smt::bpbd::server::lm::dictionary::__optimizing_↩ word_index::word_index_bucket_entry
m_len, 382
m_word, 382
m_word_id, 382
word_index_bucket_entry, 381
uva::smt::bpbd::server::lm::dictionary::__optimizing↩ _word_index::word_index_bucket_entry< word_id_type >, 381
uva::smt::bpbd::server::lm::dictionary::aword_index, 89
~aword_index, 90
count_word, 90
do_post_actions, 90
do_post_word_count, 91
get_number_of_words, 91
get_word_id, 91
is_post_actions_needed, 91
is_word_counts_needed, 91
is_word_index_continuous, 91
is_word_registering_needed, 92
register_word, 92
reserve, 92
uva::smt::bpbd::server::lm::dictionary::basic_word_↩ index, 92
~basic_word_index, 94
basic_word_index, 94
begin, 95
count_word, 95
do_post_actions, 95
do_post_word_count, 95
end, 95
get_number_of_words, 95
get_word_id, 96
is_post_actions_needed, 96
is_word_counts_needed, 96
is_word_index_continuous, 96
is_word_registering_needed, 96
m_next_new_word_id, 97
m_word_index_alloc_ptr, 97
m_word_index_map_ptr, 97
m_word_index_mem_factor, 97
register_word, 97
reserve, 97
TWordIndexAllocator, 94
TWordIndexEntry, 94
TWordIndexMap, 94
TWordIndexMapConstIter, 94
uva::smt::bpbd::server::lm::dictionary::counting_word↩ _index, 131
count_word, 133
counting_word_index, 132
do_post_actions, 133
do_post_word_count, 133
is_post_actions_needed, 133
is_word_counts_needed, 133
is_word_index_continuous, 133
is_word_registering_needed, 134
register_word, 134
uva::smt::bpbd::server::lm::dictionary::hashing_word_↩ index, 173
~hashing_word_index, 173
get_number_of_words, 174
get_word_id, 174
hashing_word_index, 173
is_post_actions_needed, 174
is_word_counts_needed, 174
is_word_index_continuous, 174
is_word_registering_needed, 174
register_word, 175
reserve, 175
uva::smt::bpbd::server::lm::dictionary::optimizing_↩ word_index
~optimizing_word_index, 228
count_word, 228
do_post_actions, 229
do_post_word_count, 229
get_number_of_words, 229
get_word_id, 229
is_post_actions_needed, 229
is_word_counts_needed, 230

is_word_index_continuous, 230
is_word_registering_needed, 230
optimizing_word_index, 228
register_word, 230
reserve, 231
uva::smt::bpbd::server::lm::dictionary::optimizing_word_index< sub_word_index_type >, 227
uva::smt::bpbd::server::lm::g2d_map_trie
~g2d_map_trie, 152
add_m_gram, 153
BASE, 152
g2d_map_trie, 152
get_m_gram_payload, 153
get_n_gram_payload, 153
get_unigram_payload, 153
get_unk_word_prob, 153
log_model_type_info, 154
pre_allocate, 154
set_def_unk_word_prob, 154
T_M_Gram_PB_Entry, 152
T_M_Gram_Prob_Entry, 152
uva::smt::bpbd::server::lm::g2d_map_trie< WordIndexType >, 151
uva::smt::bpbd::server::lm::generic_trie_base
~generic_trie_base, 156
add_m_gram, 156
BASE, 155
execute, 156
FIRST_VALID_CTX_ID, 160
generic_trie_base, 156
get_m_gram_payload, 156
get_n_gram_payload, 158
get_unigram_payload, 158
get_unk_word_prob, 158
is_context_needed, 158
is_m_gram_potentially_present, 158
log_model_type_info, 160
MGRAM_IDX_OFFSET, 160
N_GRAM_IDX_IN_M_N_ARR, 160
NEEDS_BITMAP_HASH_CACHE, 161
NUM_M_GRAM_LEVELS, 161
NUM_M_N_GRAM_LEVELS, 161
pre_allocate, 160
register_m_gram_cache, 160
UNDEFINED_ARR_IDX, 161
uva::smt::bpbd::server::lm::generic_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >, 154
uva::smt::bpbd::server::lm::h2d_map_trie
~h2d_map_trie, 171
add_m_gram, 171
BASE, 170
get_m_gram_payload, 171
get_n_gram_payload, 171
get_unigram_payload, 172
get_unk_word_prob, 172
h2d_map_trie, 170
log_model_type_info, 172
pre_allocate, 172
set_def_unk_word_prob, 172
T_M_Gram_PB_Entry, 170
T_M_Gram_Prob_Entry, 170
uva::smt::bpbd::server::lm::h2d_map_trie< WordIndexType >, 169
uva::smt::bpbd::server::lm::identifiers, 65
TLongId, 65
TShortId, 65
uva::smt::bpbd::server::lm::layered_trie_base
BASE, 177
ensure_context, 179
get_cached_context_id, 179
get_ctx_id, 179
is_context_needed, 180
layered_trie_base, 178
pre_allocate, 180
set_cache_context_id, 180
uva::smt::bpbd::server::lm::layered_trie_base< TrieType, WordIndexType, BITMAP_HASH_CACHE_BUCKETS_FACTOR >, 176
uva::smt::bpbd::server::lm::lm_configurator, 182
allocate_fast_query_proxy, 182
allocate_slow_query_proxy, 182
connect, 182
disconnect, 183
dispose_fast_query_proxy, 183
dispose_slow_query_proxy, 183
uva::smt::bpbd::server::lm::lm_parameters, 195
finalize, 195
get_lm_weight, 195
is_lm_weight, 196
m_conn_string, 196
m_lambdas, 196
m_num_lambdas, 196
uva::smt::bpbd::server::lm::m_gram_query, 210
get_curr_begin_word_id, 213
get_curr_ctx_ref, 213
get_curr_end_word_id, 213
get_curr_level, 213
get_curr_level_m1, 213
get_curr_level_m2, 213
get_curr_m_gram_hash, 214
get_curr_m_gram_id, 214
get_curr_payload_ref, 214
get_curr_uni_gram_word_id, 214
get_query_begin_word_idx, 214
get_query_end_word_idx, 215
is_curr_uni_gram, 215
is_not_finished, 215
m_curr_begin_word_idx, 217
m_curr_end_word_idx, 217
m_gram_query, 212
m_probs, 217
operator<<, 216
operator[], 215
payload_ptr, 211
set_curr_payload, 215

set_data, 216
set_word_indxes, 216
uva::smt::bpbd::server::lm::m_grams, 65
m_gram_payload, 66
operator<<, 66
uva::smt::bpbd::server::lm::m_grams::m_gram_id, 66
TM_Gram_Id_Value_Ptr, 67
uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id
allocate_byte_m_gram_id, 103
compare, 103
compute_m_gram_id, 105
create_m_gram_id, 105
gram_id_byte_len_2_type, 106
gram_id_type_2_byte_len, 106
ID_TYPE_LEN_BYTES, 109
is_equal_m_grams_id, 106
is_less_m_grams_id, 108
is_more_m_grams_id, 108
LEVEL_2_GRAM_TO_BYTE_LEN, 109
LEVEL_2_GRAM_TO_TYPE_LEN, 109
LEVEL_3_GRAM_TO_BYTE_LEN, 110
LEVEL_3_GRAM_TO_TYPE_LEN, 110
LEVEL_4_GRAM_TO_BYTE_LEN, 110
LEVEL_4_GRAM_TO_TYPE_LEN, 111
LEVEL_5_GRAM_TO_BYTE_LEN, 111
LEVEL_5_GRAM_TO_TYPE_LEN, 111
LEVEL_6_GRAM_TO_BYTE_LEN, 112
LEVEL_6_GRAM_TO_TYPE_LEN, 113
MAX_ID_LEN_BYTES, 114
NUM_BYTES_WORD_ID, 114
NUMBER_ID_TYPES_PER_LEVEL, 114
uva::smt::bpbd::server::lm::m_grams::m_gram_id::Byte_M_Gram_Id< TWordIdType >, 99
uva::smt::bpbd::server::lm::m_grams::m_gram_id::T_Gram_Id_Key, 288
m_id, 288
m_len_bytes, 288
uva::smt::bpbd::server::lm::m_grams::m_gram_payload_s, 209
m_back, 210
m_gram_payload_s, 210
m_prob, 210
uva::smt::bpbd::server::lm::m_grams::model_m_gram, 223
BASE, 224
get_hash, 224
get_next_new_token, 224
is_unk_unigram, 224
m_back_off, 225
m_payload, 225
m_prob, 225
model_m_gram, 224
operator<<, 225
prepare_for_adding, 225
start_new_m_gram, 225
uva::smt::bpbd::server::lm::m_grams::phrase_base
create_phrase_id, 232
get_first_word_idx, 234
get_last_word_id, 234
get_last_word_idx, 234
get_num_words, 234
get_phrase_id_ref, 234
m_gram_id_type, 232
operator[], 235
phrase_base, 232
set_word_ids, 235
word_ids, 235
uva::smt::bpbd::server::lm::m_grams::phrase_base< MAX_PHRASE_LENGTH, MAX_PHRASE_ID_LENGTH >, 231
uva::smt::bpbd::server::lm::m_grams::query_m_gram, 237
BASE, 238
get_hash, 238
operator<<, 238
query_m_gram, 238
set_m_gram, 238
uva::smt::bpbd::server::lm::proxy, 67
uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy, 184
~lm_fast_query_proxy, 184
execute, 185
get_begin_tag_uid, 185
get_end_tag_uid, 185
get_unk_word_prob, 186
get_word_ids, 186
uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local
~lm_fast_query_proxy_local, 188
execute, 188
get_begin_tag_uid, 188
get_end_tag_uid, 188
get_m_gram_str, 189
get_query_str, 189
get_report_interm_results, 189
get_unk_word_prob, 189
get_word_ids, 190
lm_fast_query_proxy_local, 187
report_final_result, 190
word_index_type, 187
uva::smt::bpbd::server::lm::proxy::lm_fast_query_proxy_local< trie_type >, 186
uva::smt::bpbd::server::lm::proxy::lm_proxy, 196
~lm_proxy, 197
allocate_fast_query_proxy, 197
allocate_slow_query_proxy, 197
connect, 197
disconnect, 198
dispose_fast_query_proxy, 198
dispose_slow_query_proxy, 198
uva::smt::bpbd::server::lm::proxy::lm_proxy_local, 198
~lm_proxy_local, 199
allocate_fast_query_proxy, 199
allocate_slow_query_proxy, 199
connect, 200

disconnect, 200
dispose_fast_query_proxy, 200
dispose_slow_query_proxy, 200
lm_proxy_local, 199
m_begin_tag_uid, 201
m_end_tag_uid, 201
m_model, 201
m_unk_word_prob, 201
m_word_index, 201
uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy, 201
~lm_slow_query_proxy, 202
execute, 202
uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local
~lm_slow_query_proxy_local, 203
execute, 203
get_m_gram_str, 205
get_query_str, 205
get_report_interm_results, 205
lm_slow_query_proxy_local, 203
report_final_result, 205
set_tokens_and_word_ids, 206
word_index_type, 203
uva::smt::bpbd::server::lm::proxy::lm_slow_query_proxy_local< trie_type >, 202
uva::smt::bpbd::server::lm::w2c_array_trie
~w2c_array_trie, 371
add_m_gram, 372
BASE, 371
get_ctx_id, 372
get_m_gram_payload, 372
get_n_gram_payload, 372
get_unigram_payload, 372
get_unk_word_prob, 373
is_post_grams, 373
log_model_type_info, 373
post_M_N_Grams, 373
post_grams, 373
post_m_grams, 373
post_n_grams, 374
pre_allocate, 374
set_def_unk_word_prob, 374
T_M_GramWordEntry, 371
T_N_GramWordEntry, 371
w2c_array_trie, 371
uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >, 370
uva::smt::bpbd::server::lm::w2c_array_trie< WordIndexType >::WordDataEntry< ARRAY_LEM_TYPE >, 384
uva::smt::bpbd::server::lm::w2c_array_trie::WordDataEntry
cio, 385
uva::smt::bpbd::server::lm::w2c_hybrid_trie
~w2c_hybrid_trie, 376
add_m_gram, 376
BASE, 375
get_ctx_id, 376
get_m_gram_payload, 377
get_n_gram_payload, 377
get_unigram_payload, 377
get_unk_word_prob, 377
log_model_type_info, 377
pre_allocate, 378
set_def_unk_word_prob, 378
w2c_hybrid_trie, 376
uva::smt::bpbd::server::lm::w2c_hybrid_trie< WordIndexType, StorageFactory, StorageContainer >, 374
uva::smt::bpbd::server::lm::word_index_trie_base
get_word_index, 383
is_post_grams, 383
m_word_index, 384
post_grams, 383
pre_allocate, 384
set_def_unk_word_prob, 384
word_index_trie_base, 383
WordIndexType, 383
uva::smt::bpbd::server::lm::word_index_trie_base< WordIndex >, 382
uva::smt::bpbd::server::rm, 67
rm_builder_type, 67
rm_model_reader, 67
rm_model_type, 68
uva::smt::bpbd::server::rm::builders, 68
uva::smt::bpbd::server::rm::builders::rm_basic_builder
build, 240
count_source_target_phrases, 240
parse_rm_file, 241
process_entry_weights, 241
process_source_entries, 241
rm_basic_builder, 240
uva::smt::bpbd::server::rm::builders::rm_basic_builder< model_type, reader_type >, 239
uva::smt::bpbd::server::rm::models, 68
DISCONT_LEFT_ORIENT, 69
DISCONT_RIGHT_ORIENT, 69
MONOTONE_ORIENT, 69
reordering_orientation, 68
rm_entry, 68
SWAP_ORIENT, 69
size, 69
UNKNOWN_ORIENT, 69
uva::smt::bpbd::server::rm::models::__rm_basic_model, 69
uva::smt::bpbd::server::rm::models::rm_basic_model, 241
~rm_basic_model, 242
add_entry, 243
BEGIN_SENT_TAG_UID, 247
END_SENT_TAG_UID, 247
find_begin_end_entries, 244
find_unk_entry, 244
get_begin_tag_entry, 244
get_end_tag_entry, 244

get_entry, 244, 245
is_num_entries_needed, 245
is_unk_entry, 245
log_model_type_info, 245
rm_basic_model, 242
rm_entry_map, 242
SOURCE_UNK_UID, 247
set_num_entries, 245
TARGET_UNK_UID, 247
uva::smt::bpbd::server::rm::models::rm_entry_temp
~rm_entry_temp, 249
get_weight, 249
get_weights, 249
NUM_FEATURES, 251
operator<<, 251
operator==, 250
operator[], 250
rm_entry_temp, 249
set_entry_uid, 250
uva::smt::bpbd::server::rm::models::rm_entry_temp< num_features >, 248
uva::smt::bpbd::server::rm::models::rm_query
~rm_query, 256
execute, 256
get_reordering, 257
query_map, 256
rm_query, 256
uva::smt::bpbd::server::rm::models::rm_query< model_type >, 255
uva::smt::bpbd::server::rm::proxy, 69
uva::smt::bpbd::server::rm::proxy::rm_proxy, 252
~rm_proxy, 253
allocate_query_proxy, 253
connect, 253
disconnect, 253
dispose_query_proxy, 253
uva::smt::bpbd::server::rm::proxy::rm_proxy_local, 253
~rm_proxy_local, 254
allocate_query_proxy, 254
connect, 254
disconnect, 255
dispose_query_proxy, 255
load_model_data, 255
rm_proxy_local, 254
uva::smt::bpbd::server::rm::proxy::rm_query_proxy, 257
~rm_query_proxy, 257
execute, 258
get_begin_tag_reordering, 258
get_end_tag_reordering, 258
get_reordering, 258
uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local
~rm_query_proxy_local, 259
execute, 260
get_begin_tag_reordering, 260
get_end_tag_reordering, 260
get_reordering, 260
rm_query_proxy_local, 259
uva::smt::bpbd::server::rm::proxy::rm_query_proxy_local< model_type >, 259
uva::smt::bpbd::server::rm::rm_configurator, 247
allocate_query_proxy, 247
connect, 248
disconnect, 248
dispose_query_proxy, 248
uva::smt::bpbd::server::rm::rm_parameters, 251
finalize, 252
m_conn_string, 252
m_lambdas, 252
m_num_lambdas, 252
uva::smt::bpbd::server::server_parameters, 267
m_de_params, 268
m_lm_params, 268
m_num_threads, 268
m_rm_params, 268
m_server_port, 268
m_source_lang, 268
m_target_lang, 268
m_tm_params, 269
verify, 268
uva::smt::bpbd::server::task_id, 69
uva::smt::bpbd::server::tm, 69
tm_builder_type, 69
tm_model_reader, 69
tm_model_type, 70
uva::smt::bpbd::server::tm::builders, 70
sizes_map, 70
uva::smt::bpbd::server::tm::builders::tm_basic_builder
~tm_basic_builder, 298
add_unk_translation, 299
build, 299
count_source_phrases, 299
is_good_features, 299
parse_tm_file, 299
post_process_feature, 300
process_features, 300
process_source_entries, 300
process_target_entry, 300
tm_basic_builder, 298
uva::smt::bpbd::server::tm::builders::tm_basic_builder< model_type, reader_type >, 297
uva::smt::bpbd::server::tm::models, 70
feature_array, 71
tm_const_source_entry, 71
tm_const_source_entry_ptr, 71
tm_const_target_entry, 71
tm_source_entry_ptr, 71
tm_target_entry, 71
uva::smt::bpbd::server::tm::models::__tm_basic_model, 71
uva::smt::bpbd::server::tm::models::tm_basic_model, 301
~tm_basic_model, 302
begin_entry, 302
finalize, 302
finalize_entry, 302

get_source_entry, 302
is_num_entries_needed, 304
is_unk_entry, 304
log_model_type_info, 304
set_num_entries, 304
set_unk_entry, 304
tm_basic_model, 302
tm_source_entry_map, 302
uva::smt::bpbd::server::tm::models::tm_query
~tm_query, 313
execute, 313
get_source_entry, 313
get_st_uids, 313
query_map, 312
tm_query, 313
uva::smt::bpbd::server::tm::models::tm_query< model_type >, 312
uva::smt::bpbd::server::tm::models::tm_source_entry, 317
~tm_source_entry, 317
add_target, 318
begin, 318
finalize, 318
get_min_cost, 318
get_source_uid, 318
get_st_uids, 319
get_targets, 319
has_target, 319
has_translations, 319
num_targets, 319
operator==, 320
set_source_uid, 320
tm_source_entry, 317
uva::smt::bpbd::server::tm::models::tm_target_entry_temp
~tm_target_entry_temp, 321
get_num_words, 322
get_st_uid, 322
get_t_c_s, 322
get_target_phrase, 322
get_total_weight, 322
get_word_ids, 323
is_unk_trans, 323
NUM_FEATURES, 324
set_data, 323
set_features, 323
tm_target_entry_temp, 321
UNKNOWN_TARGET_ENTRY_UID, 324
uva::smt::bpbd::server::tm::models::tm_target_entry_temp< max_num_features >, 320
uva::smt::bpbd::server::tm::proxy, 71
uva::smt::bpbd::server::tm::proxy::tm_proxy, 308
~tm_proxy, 309
allocate_query_proxy, 309
connect, 309
disconnect, 309
dispose_query_proxy, 310
uva::smt::bpbd::server::tm::proxy::tm_proxy_local, 310
~tm_proxy_local, 311
allocate_query_proxy, 311
connect, 311
disconnect, 311
dispose_query_proxy, 311
load_model_data, 311
tm_proxy_local, 311
uva::smt::bpbd::server::tm::proxy::tm_query_proxy, 314
~tm_query_proxy, 314
execute, 314
get_source_entry, 315
get_st_uids, 315
uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local
~tm_query_proxy_local, 316
execute, 316
get_source_entry, 316
get_st_uids, 316
tm_query_proxy_local, 316
uva::smt::bpbd::server::tm::proxy::tm_query_proxy_local< model_type >, 315
uva::smt::bpbd::server::tm::tm_configurator, 306
allocate_query_proxy, 306
connect, 306
disconnect, 307
dispose_query_proxy, 307
uva::smt::bpbd::server::tm::tm_parameters, 307
finalize, 308
m_conn_string, 308
m_lambdas, 308
m_min_tran_prob, 308
m_num_lambdas, 308
m_num_unk_features, 308
m_trans_limit, 308
m_unk_features, 308
uva::smt::bpbd::server::trans_job, 325
~trans_job, 327
cancel, 327
combine_job_result, 327
done_job_notifier, 326
get_code, 327
get_job_id, 327
get_session_id, 327
get_tasks, 327
get_text, 328
is_job_finished, 328
notify_task_done, 328
set_done_job_notifier, 328
tasks_const_iter_type, 326
tasks_iter_type, 326
tasks_list_type, 326
trans_job, 326
uva::smt::bpbd::server::trans_job_pool, 331
~trans_job_pool, 333
add_job, 333
cancel_all_jobs, 333
cancel_jobs, 333
delete_job, 334

finished_job_notifier, 332
is_stop_running, 334
jobs_list_iter_type, 332
jobs_list_type, 332
jobs_map_iter_type, 332
jobs_map_type, 332
notify_job_done, 334
plan_new_job, 334
process_finished_jobs, 334
report_run_time_info, 334
sessions_map_iter_type, 332
sessions_map_type, 333
set_job_result_setter, 334
set_num_threads, 336
stop, 336
trans_job_pool, 333
wake_up_jobs_thread, 336
uva::smt::bpbd::server::trans_manager, 344
~trans_manager, 346
close_session, 346
handlers_map_iter_type, 345
handlers_map_type, 345
notify_job_finished, 346
open_session, 346
report_run_time_info, 346
response_sender, 345
sessions_map_type, 345
set_num_threads, 346
set_response_sender, 347
stop, 347
trans_manager, 345
translate, 347
uva::smt::bpbd::server::trans_task, 351
~trans_task, 353
cancel, 353
cancel_task_notifier, 351
done_task_notifier, 351
get_code, 353
get_source_text, 353
get_target_text, 353
get_task_id, 353
process_task_result, 354
set_cancel_task_notifier, 354
trans_task, 352
translate, 354
uva::smt::bpbd::server::trans_task_pool, 354
~trans_task_pool, 356
m_condition, 357
m_queue_mutex, 357
m_stop, 357
m_tasks, 357
notify_task_cancel, 356
plan_new_task, 356
report_run_time_info, 356
set_num_threads, 356
tasks_queue_iter_type, 355
tasks_queue_type, 355
threads_list_type, 355
trans_task_pool, 355
trans_task_pool_worker, 357
workers_list_type, 355
uva::smt::bpbd::server::trans_task_pool_worker, 357
~trans_task_pool_worker, 358
is_busy, 358
operator(), 358
stop, 358
trans_task_pool_worker, 358
uva::smt::bpbd::server::translation_server, 361
on_close, 362
on_fail, 363
on_message, 363
on_open, 363
report_run_time_info, 363
run, 363
send_response, 363
server, 362
set_num_threads, 364
stop, 364
translation_server, 362
uva::utils, 71
uva::utils::containers, 72
_memIncTypesEnumStr, 73
CONSTANT, 73
get_mem_incr_strat, 73
LINEAR, 73
LOG_10, 73
LOG_2, 73
mem_inc_types_enum, 73
size, 73
TCapacityIncFunct, 72
UNDEFINED, 73
uva::utils::containers::ELEMENT_DEALLOC_FUNC
func_ptr, 145
func_type, 145
NULL_FUNC_PTR, 145
uva::utils::containers::ELEMENT_DEALLOC_FUNC< ELEM_TYPE >, 144
uva::utils::containers::alloc, 74
allocate_container, 74
deallocate_container, 74
operator!=, 75
operator==, 75
reserve_mem_unordered_map, 75
uva::utils::containers::alloc::greedy_memory_allocator_manager, 166
~greedy_memory_allocator, 163
address, 164
allocate, 164
available, 164
const_pointer, 162
const_reference, 162
construct, 164
deallocate, 166
destroy, 166
difference_type, 162
getStorageRef, 166

greedy_memory_allocator, 163
max_size, 166
pointer, 162
reference, 163
size_type, 163
value_type, 163
uva::utils::containers::alloc::greedy_memory_allocator< T >, 161
uva::utils::containers::alloc::greedy_memory_allocator< T >::rebind< U >, 239
uva::utils::containers::alloc::greedy_memory_allocator::rebind
other, 239
uva::utils::containers::circular_queue
~circular_queue, 127
circular_queue, 127
empty_queue, 127
get_capacity, 128
get_elems, 128
get_size, 128
is_equal_last, 128
push_back, 128, 130
tail_to_string, 130
uva::utils::containers::circular_queue< elem_type, capacity >, 126
uva::utils::containers::dynamic_stack_array
~dynamic_stack_array, 142
allocate, 142
data, 142
dynamic_stack_array, 142
ELEMENT_TYPE_PTR, 141
has_data, 142
MAX_SIZE_TYPE_VALUE, 144
operator[], 143
PARAMETERS_SIZE_BYTES, 144
pre_allocate, 143
shrink, 143
size, 143
sort, 144
TElemType, 141
TIndexType, 141
uva::utils::containers::dynamic_stack_array< ELEMENT_TYPE, IDX_DATA_TYPE, INITIAL_CAPACITY, DESTRUCTOR >, 140
uva::utils::containers::fixed_size_hashmap
~fixed_size_hashmap, 150
add_new_element, 150
fixed_size_hashmap, 148
get_element, 150
MAX_ELEMENT_INDEX, 150
MIN_ELEMENT_INDEX, 151
NO_ELEMENT_INDEX, 151
TElemType, 148
uva::utils::containers::fixed_size_hashmap< ELEMENT_TYPE, KEY_TYPE, IDX_TYPE >, 147
uva::utils::containers::greedy_memory_storage, 167
_allocBytes, 169
_memoryBuffers, 169
_numBytes, 169
_pBuffer, 169
~greedy_memory_storage, 168
allocate, 168
getAvailableBytes, 168
getBufferSizeBytes, 168
greedy_memory_storage, 168
size_type, 167
TStorageData, 167
uva::utils::containers::mem_increase_strategy, 217
get_new_capacity, 219
get_strategy_info, 219
mem_increase_strategy, 218, 219
uva::utils::containers::upp_diag_matrix
~upp_diag_matrix, 367
element_type_ptr, 367
get_dim, 367
m_max_idx, 368
m_min_idx, 368
operator[], 367
upp_diag_matrix, 367
uva::utils::containers::upp_diag_matrix< element_type >, 366
uva::utils::containers::utils, 75
is_less, 76
my_bsearch, 76
my_bsearch_id, 77
my_bsearch_wordId_ctxId, 78
my_isearch_id, 78
my_lsearch_id, 79
my_sort, 79, 80
uva::utils::containers::utils::T_IS_COMPARE_FUNC
func_ptr, 289
func_type, 289
uva::utils::containers::utils::T_IS_COMPARE_FUNC< ELEM_TYPE >, 288
uva::utils::exceptions, 80
DO_SANITY_CHECKS, 80
uva::utils::exceptions::uva_exception, 368
~uva_exception, 369
get_message, 369
uva_exception, 369
what, 369
uva::utils::file, 80
operator<<, 81
tokens_to_string, 81
uva::utils::file::afile_reader, 87
~afile_reader, 88
afile_reader, 88
close, 88
get_first, 88
get_first_line, 88
get_first_space, 88
get_first_tab, 88
get_last, 88
get_last_space, 88
is_open, 89
log_reader_type_info, 89

operator bool, 89
reset, 89
uva::utils::file::cstyle_file_reader, 134
~cstyle_file_reader, 135
close, 135
cstyle_file_reader, 135
get_first_line, 135
is_open, 136
log_reader_type_info, 136
operator bool, 136
reset, 136
uva::utils::file::file_stream_reader, 145
~file_stream_reader, 146
close, 146
file_stream_reader, 146
get_first_line, 146
is_open, 147
log_reader_type_info, 147
operator bool, 147
reset, 147
uva::utils::file::memory_mapped_file_reader, 219
close, 221
get_first_line, 221
is_open, 221
log_reader_type_info, 221
memory_mapped_file_reader, 221
operator bool, 221
uva::utils::file::text_piece_reader, 290
copy_string, 292
find_first_subseq, 292
get_begin_c_str, 292
get_begin_ptr, 292
get_first, 292
get_first_line, 293
get_first_space, 293
get_first_tab, 293
get_last, 293
get_last_space, 295
get_rest_c_str, 295
get_rest_str, 295
has_more, 295
length, 295
operator!=, 296
operator==, 296, 297
operator[], 297
set, 297
str, 297
text_piece_reader, 291
uva::utils::hashing, 81
uva::utils::logging, 81
DEBUG, 82
DEBUG1, 82
DEBUG2, 82
DEBUG3, 82
DEBUG4, 82
debug_levels_enum, 82
ERROR, 82
INFO, 82
INFO1, 82
INFO2, 82
INFO3, 82
operator<<, 82
RESULT, 82
size, 82
USAGE, 82
WARNING, 82
uva::utils::logging::logger, 206
~logger, 206
get, 206, 207
get_curr_level_str, 207
get_reporting_level, 207
get_reporting_levels, 207
is_progress_bar_on, 207
is_relevant_level, 208
set_reporting_level, 208
start_progress_bar, 208
stop_progress_bar, 208
update_progress_bar, 208
uva::utils::logging::logging_synch, 209
mv, 209
rec_scoped_lock, 209
uva::utils::math, 83
uva::utils::math::bits, 83
uva::utils::math::const_expr, 83
ceil, 83
log2, 83
power, 83
uva::utils::math::log2, 83
uva::utils::monitor, 83
BYTES_ONE_MB, 84
TMemotyUsage, 84
uva::utils::monitor::memory_usage, 222
memory_usage, 222
vmhwm, 222
vmpeak, 222
vmrss, 222
vmsize, 223
uva::utils::monitor::stat_monitor, 283
get_cpu_time, 283
get_mem_stat, 283
uva::utils::text, 84
ASCII_SPACE_CHAR, 84
UTF8_ASCII_PUNCTUATIONS, 84
UTF8_ASCII_WHITESPACES, 84
UTF8_EMPTY_STRING, 84
UTF8_NEW_LINE_STRING, 84
UTF8_SPACE_STRING, 84
uva::utils::threads, 85
a_bool_flag, 85
acr_bool_flag, 85
recursive_guard, 85
scoped_guard, 85
unique_guard, 85
uva_exception
uva::utils::exceptions::uva_exception, 369

VALUE_LEN_BYTES

math_utils.hpp, 411
val
uva::smt::bpbd::common::messaging::trans_job↩
_code, 331
valid_digit
string_utils.hpp, 413
value_type
uva::utils::containers::alloc::greedy_memory_↩
allocator, 163
values
uva::smt::bpbd::client::trans_job_status, 342
uva::smt::bpbd::common::messaging::trans_job↩
_code, 329
verify
uva::smt::bpbd::server::server_parameters, 268
vmhwm
uva::utils::monitor::memory_usage, 222
vmpeak
uva::utils::monitor::memory_usage, 222
vmrss
uva::utils::monitor::memory_usage, 222
vmsize
uva::utils::monitor::memory_usage, 223

W2CH_UM_Storage
uva::smt::bpbd::server::lm::W2CH_UM_Storage,
379
W2CH_UM_StorageFactory
uva::smt::bpbd::server::lm::W2CH_UM_Storage↩
Factory, 380
w2c_array_trie
uva::smt::bpbd::server::lm::w2c_array_trie, 371
w2c_hybrid_trie
uva::smt::bpbd::server::lm::w2c_hybrid_trie, 376
WARNING
uva::utils::logging, 82
WARNING_PARAM_VALUE
logger.hpp, 410
WHITE_SPACE_SEPARATOR
logger.hpp, 410
wait
uva::smt::bpbd::client::trans_manager, 350
wait_connect
uva::smt::bpbd::client::translation_client, 361
wake_up_jobs_thread
uva::smt::bpbd::server::trans_job_pool, 336
what
uva::utils::exceptions::uva_exception, 369
word
uva::smt::bpbd::server::lm::dictionary::__counting↩
_word_index::TWordInfo, 366
word_id
uva::smt::bpbd::server::lm::__C2WArrayTrie::T↩
CtxIdProbData, 290
word_ids
uva::smt::bpbd::server::lm::m_grams::phrase_↩
base, 235
word_index_bucket_entry
optimizing_word_index.hpp, 425
uva::smt::bpbd::server::lm::dictionary::__optimizing↩
_word_index::word_index_bucket_entry, 381
word_index_trie_base
uva::smt::bpbd::server::lm::word_index_trie_base,
383
word_index_type
uva::smt::bpbd::server::lm::proxy::lm_fast_query↩
_proxy_local, 187
uva::smt::bpbd::server::lm::proxy::lm_slow_↩
query_proxy_local, 203
word_uid
uva::smt::bpbd::server, 42
WordIndexType
uva::smt::bpbd::server::lm::arpa::lm_basic_builder,
181
uva::smt::bpbd::server::lm::arpa::lm_gram_↩
builder_factory, 194
uva::smt::bpbd::server::lm::word_index_trie_base,
383
workers_list_type
uva::smt::bpbd::server::trans_task_pool, 355
write_received_job_result
uva::smt::bpbd::client::trans_manager, 350
write_result_to_file
uva::smt::bpbd::client::trans_manager, 350

ZERRO_WORD_IDX
uva::smt::bpbd::server::decoder::stack::state_↩
data_templ, 287