

FastqArazketa

Generated by Doxygen 1.8.8

Mon Oct 9 2017 11:40:10

Contents

| | | |
|----------|---|----------|
| 1 | Class Index | 1 |
| 1.1 | Class List | 1 |
| 2 | File Index | 3 |
| 2.1 | File List | 3 |
| 3 | Class Documentation | 7 |
| 3.1 | _ad_seq Struct Reference | 7 |
| 3.1.1 | Detailed Description | 7 |
| 3.1.2 | Member Data Documentation | 7 |
| 3.1.2.1 | L | 7 |
| 3.1.2.2 | Lpack | 7 |
| 3.1.2.3 | Lpack_sh | 7 |
| 3.1.2.4 | pack | 7 |
| 3.1.2.5 | pack_sh | 8 |
| 3.1.2.6 | seq | 8 |
| 3.2 | _adapter Struct Reference | 8 |
| 3.2.1 | Detailed Description | 8 |
| 3.2.2 | Member Data Documentation | 8 |
| 3.2.2.1 | ad2_fa | 8 |
| 3.2.2.2 | ad_fa | 8 |
| 3.2.2.3 | mismatches | 8 |
| 3.2.2.4 | Nad | 8 |
| 3.2.2.5 | threshold | 9 |
| 3.3 | _bfilter Struct Reference | 9 |
| 3.3.1 | Detailed Description | 9 |
| 3.3.2 | Member Data Documentation | 9 |
| 3.3.2.1 | bfszBits | 9 |
| 3.3.2.2 | bfszBytes | 9 |
| 3.3.2.3 | falsePosRate | 9 |
| 3.3.2.4 | filter | 9 |
| 3.3.2.5 | hashNum | 9 |

| | | |
|---------|--|----|
| 3.3.2.6 | kmersize | 10 |
| 3.3.2.7 | kmersizeBytes | 10 |
| 3.3.2.8 | nelem | 10 |
| 3.4 | _bfkmer Struct Reference | 10 |
| 3.4.1 | Detailed Description | 10 |
| 3.4.2 | Member Data Documentation | 10 |
| 3.4.2.1 | compact | 10 |
| 3.4.2.2 | halfsizeBytes | 10 |
| 3.4.2.3 | hangingBases | 10 |
| 3.4.2.4 | hashNum | 11 |
| 3.4.2.5 | hashValues | 11 |
| 3.4.2.6 | hasOverhead | 11 |
| 3.4.2.7 | kmersize | 11 |
| 3.4.2.8 | kmersizeBytes | 11 |
| 3.5 | _ds_adap Struct Reference | 11 |
| 3.5.1 | Detailed Description | 11 |
| 3.5.2 | Member Data Documentation | 11 |
| 3.5.2.1 | ad1 | 11 |
| 3.5.2.2 | ad2 | 11 |
| 3.5.2.3 | L1 | 12 |
| 3.5.2.4 | L2 | 12 |
| 3.6 | _fa_data Struct Reference | 12 |
| 3.6.1 | Detailed Description | 12 |
| 3.6.2 | Member Data Documentation | 12 |
| 3.6.2.1 | entry | 12 |
| 3.6.2.2 | entrylen | 13 |
| 3.6.2.3 | linelen | 13 |
| 3.6.2.4 | nentries | 13 |
| 3.6.2.5 | nlines | 13 |
| 3.7 | _fa_entry Struct Reference | 13 |
| 3.7.1 | Detailed Description | 13 |
| 3.7.2 | Member Data Documentation | 13 |
| 3.7.2.1 | N | 13 |
| 3.7.2.2 | seq | 13 |
| 3.8 | _fq_read Struct Reference | 14 |
| 3.8.1 | Detailed Description | 14 |
| 3.8.2 | Member Data Documentation | 14 |
| 3.8.2.1 | extended | 14 |
| 3.8.2.2 | L | 14 |
| 3.8.2.3 | L_ad | 14 |

| | | |
|----------|--|----|
| 3.8.2.4 | L_ext | 14 |
| 3.8.2.5 | L_pack | 14 |
| 3.8.2.6 | L_packsh | 14 |
| 3.8.2.7 | Lhalf | 15 |
| 3.8.2.8 | line1 | 15 |
| 3.8.2.9 | line2 | 15 |
| 3.8.2.10 | line3 | 15 |
| 3.8.2.11 | line4 | 15 |
| 3.8.2.12 | pack | 15 |
| 3.8.2.13 | packsh | 15 |
| 3.8.2.14 | start | 15 |
| 3.9 | _iparam_makeBloom Struct Reference | 15 |
| 3.9.1 | Detailed Description | 16 |
| 3.9.2 | Member Data Documentation | 16 |
| 3.9.2.1 | bfszBits | 16 |
| 3.9.2.2 | falsePosRate | 16 |
| 3.9.2.3 | filterfile | 16 |
| 3.9.2.4 | hashNum | 16 |
| 3.9.2.5 | inputfasta | 16 |
| 3.9.2.6 | kmersize | 16 |
| 3.9.2.7 | nelem | 16 |
| 3.9.2.8 | paramfile | 16 |
| 3.10 | _iparam_makeTree Struct Reference | 16 |
| 3.10.1 | Detailed Description | 17 |
| 3.10.2 | Member Data Documentation | 17 |
| 3.10.2.1 | inputfasta | 17 |
| 3.10.2.2 | L | 17 |
| 3.10.2.3 | outputfile | 17 |
| 3.11 | _iparam_Qreport Struct Reference | 17 |
| 3.11.1 | Detailed Description | 17 |
| 3.11.2 | Member Data Documentation | 18 |
| 3.11.2.1 | filter | 18 |
| 3.11.2.2 | inputfile | 18 |
| 3.11.2.3 | minQ | 18 |
| 3.11.2.4 | nQ | 18 |
| 3.11.2.5 | ntiles | 18 |
| 3.11.2.6 | one_read_len | 18 |
| 3.11.2.7 | outputfilebin | 18 |
| 3.11.2.8 | outputfilehtml | 18 |
| 3.11.2.9 | outputfileinfo | 18 |

| | |
|--|----|
| 3.11.2.10 read_len | 18 |
| 3.12 _iparam_Sreport Struct Reference | 18 |
| 3.12.1 Detailed Description | 19 |
| 3.12.2 Member Data Documentation | 19 |
| 3.12.2.1 inputfolder | 19 |
| 3.12.2.2 outputfile | 19 |
| 3.12.2.3 Rmd_file | 19 |
| 3.13 _iparam_trimFilter Struct Reference | 19 |
| 3.13.1 Detailed Description | 20 |
| 3.13.2 Member Data Documentation | 20 |
| 3.13.2.1 ad | 20 |
| 3.13.2.2 globleft | 20 |
| 3.13.2.3 globright | 20 |
| 3.13.2.4 lfa | 20 |
| 3.13.2.5 lfq | 20 |
| 3.13.2.6 lfq2 | 21 |
| 3.13.2.7 lidx | 21 |
| 3.13.2.8 linfo | 21 |
| 3.13.2.9 is_adapter | 21 |
| 3.13.2.10 is_fa | 21 |
| 3.13.2.11 is_idx | 21 |
| 3.13.2.12 kmersize | 21 |
| 3.13.2.13 L | 21 |
| 3.13.2.14 method | 21 |
| 3.13.2.15 minL | 21 |
| 3.13.2.16 minQ | 21 |
| 3.13.2.17 nlowQ | 21 |
| 3.13.2.18 Oprefix | 22 |
| 3.13.2.19 percent | 22 |
| 3.13.2.20 ptr_bfkmer | 22 |
| 3.13.2.21 score | 22 |
| 3.13.2.22 trimN | 22 |
| 3.13.2.23 trimQ | 22 |
| 3.14 _node Struct Reference | 22 |
| 3.14.1 Detailed Description | 22 |
| 3.14.2 Member Data Documentation | 23 |
| 3.14.2.1 children | 23 |
| 3.15 _split Struct Reference | 23 |
| 3.15.1 Detailed Description | 23 |
| 3.15.2 Member Data Documentation | 23 |

| | | |
|----------|------------------------------|----|
| 3.15.2.1 | N | 23 |
| 3.15.2.2 | s | 23 |
| 3.16 | _stats_TF Struct Reference | 23 |
| 3.16.1 | Detailed Description | 24 |
| 3.16.2 | Member Data Documentation | 24 |
| 3.16.2.1 | discarded | 24 |
| 3.16.2.2 | filters | 24 |
| 3.16.2.3 | good | 24 |
| 3.16.2.4 | nreads | 24 |
| 3.16.2.5 | trimmed | 24 |
| 3.17 | _stats_TFDS Struct Reference | 24 |
| 3.17.1 | Detailed Description | 24 |
| 3.17.2 | Member Data Documentation | 25 |
| 3.17.2.1 | discarded | 25 |
| 3.17.2.2 | filters | 25 |
| 3.17.2.3 | good | 25 |
| 3.17.2.4 | nreads | 25 |
| 3.17.2.5 | trimmed1 | 25 |
| 3.17.2.6 | trimmed2 | 25 |
| 3.18 | _tree Struct Reference | 25 |
| 3.18.1 | Detailed Description | 26 |
| 3.18.2 | Member Data Documentation | 26 |
| 3.18.2.1 | L | 26 |
| 3.18.2.2 | nnodes | 26 |
| 3.18.2.3 | pool_2D | 26 |
| 3.18.2.4 | pool_available | 26 |
| 3.18.2.5 | pool_count | 26 |
| 3.19 | _uint128 Struct Reference | 26 |
| 3.20 | statsinfo Struct Reference | 27 |
| 3.20.1 | Detailed Description | 27 |
| 3.20.2 | Member Data Documentation | 27 |
| 3.20.2.1 | ACGT_pos | 27 |
| 3.20.2.2 | ACGT_tile | 27 |
| 3.20.2.3 | lane_tags | 27 |
| 3.20.2.4 | lowQ_ACGT_tile | 28 |
| 3.20.2.5 | minQ | 28 |
| 3.20.2.6 | nQ | 28 |
| 3.20.2.7 | nreads | 28 |
| 3.20.2.8 | ntiles | 28 |
| 3.20.2.9 | QPosTile_table | 28 |

| | | |
|-----------|-----------------------------------|-----------|
| 3.20.2.10 | qual_tags | 28 |
| 3.20.2.11 | read_len | 28 |
| 3.20.2.12 | reads_MlowQ | 28 |
| 3.20.2.13 | reads_wN | 28 |
| 3.20.2.14 | sz_ACGT_pos | 28 |
| 3.20.2.15 | sz_ACGT_tile | 28 |
| 3.20.2.16 | sz_lowQ_ACGT_tile | 29 |
| 3.20.2.17 | sz_QPosTile_table | 29 |
| 3.20.2.18 | sz_reads_MlowQ | 29 |
| 3.20.2.19 | tile_pos | 29 |
| 3.20.2.20 | tile_tags | 29 |
| 4 | File Documentation | 31 |
| 4.1 | include/adapters.h File Reference | 31 |
| 4.1.1 | Detailed Description | 32 |
| 4.1.2 | Function Documentation | 32 |
| 4.1.2.1 | init_allUTs | 32 |
| 4.1.2.2 | obtain_score | 34 |
| 4.1.2.3 | pack_adapter | 34 |
| 4.1.2.4 | process_seq | 34 |
| 4.2 | include/bloom.h File Reference | 35 |
| 4.2.1 | Detailed Description | 36 |
| 4.2.2 | Function Documentation | 37 |
| 4.2.2.1 | compact_kmer | 37 |
| 4.2.2.2 | contains | 38 |
| 4.2.2.3 | create_Bfilter | 38 |
| 4.2.2.4 | init_Bfilter | 39 |
| 4.2.2.5 | init_Bfkmer | 39 |
| 4.2.2.6 | init_LUTs | 39 |
| 4.2.2.7 | insert_and_fetch | 40 |
| 4.2.2.8 | multiHash | 41 |
| 4.2.2.9 | read_Bfilter | 41 |
| 4.2.2.10 | save_Bfilter | 41 |
| 4.2.3 | Variable Documentation | 42 |
| 4.2.3.1 | bitMask | 42 |
| 4.3 | include/city.h File Reference | 42 |
| 4.3.1 | Detailed Description | 43 |
| 4.4 | include/citycrc.h File Reference | 43 |
| 4.4.1 | Detailed Description | 44 |
| 4.5 | include/defines.h File Reference | 44 |

| | | |
|----------|--------------------------------|----|
| 4.5.1 | Detailed Description | 46 |
| 4.5.2 | Macro Definition Documentation | 46 |
| 4.5.2.1 | ADAP | 46 |
| 4.5.2.2 | ADAP2 | 46 |
| 4.5.2.3 | ALL | 46 |
| 4.5.2.4 | B_LEN | 46 |
| 4.5.2.5 | BASESPERCHAR | 46 |
| 4.5.2.6 | BITSPERCHAR | 46 |
| 4.5.2.7 | BLOOM | 46 |
| 4.5.2.8 | bool | 46 |
| 4.5.2.9 | CONT | 46 |
| 4.5.2.10 | CONT2 | 47 |
| 4.5.2.11 | DEFAULT_MINL | 47 |
| 4.5.2.12 | DEFAULT_MINQ | 47 |
| 4.5.2.13 | DEFAULT_NQ | 47 |
| 4.5.2.14 | DEFAULT_NTILES | 47 |
| 4.5.2.15 | ENDS | 47 |
| 4.5.2.16 | ENDSFRAC | 47 |
| 4.5.2.17 | ERROR | 47 |
| 4.5.2.18 | FA_ENTRY_BUF | 47 |
| 4.5.2.19 | false | 47 |
| 4.5.2.20 | FALSE_POS_RATE | 47 |
| 4.5.2.21 | FRAC | 47 |
| 4.5.2.22 | GLOBAL | 48 |
| 4.5.2.23 | GOOD | 48 |
| 4.5.2.24 | GOOD2 | 48 |
| 4.5.2.25 | KMER_LEN | 48 |
| 4.5.2.26 | LOG_4 | 48 |
| 4.5.2.27 | LOWQ | 48 |
| 4.5.2.28 | LOWQ2 | 48 |
| 4.5.2.29 | max | 48 |
| 4.5.2.30 | MAX_FASZ_TREE | 48 |
| 4.5.2.31 | MAX_FILENAME | 48 |
| 4.5.2.32 | MAX_RCOMMAND | 48 |
| 4.5.2.33 | mem_usage | 48 |
| 4.5.2.34 | mem_usageMB | 49 |
| 4.5.2.35 | min | 49 |
| 4.5.2.36 | MIN_NMATCHES | 49 |
| 4.5.2.37 | N_ACGT | 49 |
| 4.5.2.38 | NFILES_DS | 49 |

| | | |
|----------|--|----|
| 4.5.2.39 | NFILTERS | 49 |
| 4.5.2.40 | NNNN | 49 |
| 4.5.2.41 | NNNN2 | 49 |
| 4.5.2.42 | NO | 49 |
| 4.5.2.43 | NPOOL_1D | 49 |
| 4.5.2.44 | NPOOL_2D | 49 |
| 4.5.2.45 | SA | 50 |
| 4.5.2.46 | STRIP | 50 |
| 4.5.2.47 | T_ACGT | 50 |
| 4.5.2.48 | TREE | 50 |
| 4.5.2.49 | true | 50 |
| 4.5.2.50 | ZERO_POS_RATE | 50 |
| 4.5.2.51 | ZEROQ | 50 |
| 4.6 | include/fa_read.h File Reference | 50 |
| 4.6.1 | Detailed Description | 51 |
| 4.6.2 | Function Documentation | 51 |
| 4.6.2.1 | free_fasta | 51 |
| 4.6.2.2 | nkmers | 52 |
| 4.6.2.3 | read_fasta | 52 |
| 4.6.2.4 | size_fasta | 52 |
| 4.7 | include/fopen_gen.h File Reference | 53 |
| 4.7.1 | Detailed Description | 53 |
| 4.7.2 | Function Documentation | 54 |
| 4.7.2.1 | fopen_gen | 54 |
| 4.8 | include/fq_read.h File Reference | 54 |
| 4.8.1 | Detailed Description | 55 |
| 4.8.2 | Function Documentation | 55 |
| 4.8.2.1 | get_fqread | 55 |
| 4.8.2.2 | string_seq | 56 |
| 4.9 | include/init_makeBloom.h File Reference | 56 |
| 4.9.1 | Detailed Description | 57 |
| 4.9.2 | Typedef Documentation | 57 |
| 4.9.2.1 | lparam_makeBloom | 57 |
| 4.10 | include/init_makeTree.h File Reference | 58 |
| 4.10.1 | Detailed Description | 59 |
| 4.11 | include/init_Qreport.h File Reference | 59 |
| 4.11.1 | Detailed Description | 60 |
| 4.12 | include/init_Sreport.h File Reference | 60 |
| 4.12.1 | Detailed Description | 62 |
| 4.13 | include/init_trimFilter.h File Reference | 62 |

| | |
|---|----|
| 4.13.1 Detailed Description | 63 |
| 4.14 include/init_trimFilterDS.h File Reference | 64 |
| 4.14.1 Detailed Description | 65 |
| 4.15 include/io_trimFilter.h File Reference | 65 |
| 4.15.1 Detailed Description | 66 |
| 4.15.2 Function Documentation | 66 |
| 4.15.2.1 buffer_output | 66 |
| 4.16 include/io_trimFilterDS.h File Reference | 67 |
| 4.16.1 Detailed Description | 68 |
| 4.16.2 Function Documentation | 68 |
| 4.16.2.1 buffer_outputDS | 68 |
| 4.17 include/Lmer.h File Reference | 68 |
| 4.17.1 Detailed Description | 69 |
| 4.17.2 Function Documentation | 69 |
| 4.17.2.1 init_map | 69 |
| 4.17.2.2 init_map_SA | 69 |
| 4.18 include/Rcommand_Qreport.h File Reference | 69 |
| 4.18.1 Detailed Description | 70 |
| 4.19 include/Rcommand_Sreport.h File Reference | 70 |
| 4.19.1 Detailed Description | 71 |
| 4.19.2 Function Documentation | 71 |
| 4.19.2.1 command_Sreport | 71 |
| 4.20 include/stats_info.h File Reference | 71 |
| 4.20.1 Detailed Description | 73 |
| 4.20.2 Function Documentation | 73 |
| 4.20.2.1 init_info | 73 |
| 4.20.2.2 resize_info | 73 |
| 4.20.2.3 update_ACGT_counts | 73 |
| 4.21 include/str_manip.h File Reference | 74 |
| 4.21.1 Detailed Description | 74 |
| 4.21.2 Function Documentation | 74 |
| 4.21.2.1 strindex | 74 |
| 4.21.2.2 strsplit | 75 |
| 4.22 include/struct_trimFilter.h File Reference | 75 |
| 4.22.1 Detailed Description | 76 |
| 4.22.2 Typedef Documentation | 76 |
| 4.22.2.1 Adapter | 76 |
| 4.23 include/tree.h File Reference | 77 |
| 4.23.1 Detailed Description | 78 |
| 4.23.2 Typedef Documentation | 78 |

| | | |
|----------|---------------------------------|----|
| 4.23.2.1 | Tree | 78 |
| 4.23.3 | Function Documentation | 79 |
| 4.23.3.1 | check_path | 79 |
| 4.23.3.2 | free_all_nodes | 80 |
| 4.23.3.3 | get_new_pool | 80 |
| 4.23.3.4 | new_node_buf | 80 |
| 4.23.3.5 | read_tree | 80 |
| 4.23.3.6 | save_tree | 81 |
| 4.23.3.7 | tree_from_fasta | 82 |
| 4.24 | include/trim.h File Reference | 82 |
| 4.24.1 | Detailed Description | 83 |
| 4.24.2 | Function Documentation | 83 |
| 4.24.2.1 | is_read_inBloom | 83 |
| 4.24.2.2 | is_read_inTree | 84 |
| 4.24.2.3 | Qtrim_global | 84 |
| 4.24.2.4 | trim_adapter | 84 |
| 4.24.2.5 | trim_sequenceN | 85 |
| 4.24.2.6 | trim_sequenceQ | 85 |
| 4.25 | include/trimDS.h File Reference | 85 |
| 4.25.1 | Detailed Description | 87 |
| 4.25.2 | Function Documentation | 87 |
| 4.25.2.1 | init_DSadap | 87 |
| 4.25.2.2 | trim_adapterDS | 87 |
| 4.26 | src/adapters.c File Reference | 87 |
| 4.26.1 | Detailed Description | 88 |
| 4.26.2 | Function Documentation | 89 |
| 4.26.2.1 | init_allUTs | 89 |
| 4.26.2.2 | obtain_score | 89 |
| 4.26.2.3 | pack_adapter | 89 |
| 4.26.2.4 | process_seq | 89 |
| 4.26.3 | Variable Documentation | 90 |
| 4.26.3.1 | albw0 | 90 |
| 4.26.3.2 | albw1 | 90 |
| 4.26.3.3 | alfw0 | 90 |
| 4.26.3.4 | alfw1 | 90 |
| 4.26.3.5 | bw_1B | 90 |
| 4.26.3.6 | fw_1B | 90 |
| 4.27 | src/bloom.c File Reference | 90 |
| 4.27.1 | Detailed Description | 92 |
| 4.27.2 | Function Documentation | 92 |

| | | |
|-----------|--|-----|
| 4.27.2.1 | compact_kmer | 92 |
| 4.27.2.2 | contains | 93 |
| 4.27.2.3 | create_Bfilter | 94 |
| 4.27.2.4 | init_Bfilter | 94 |
| 4.27.2.5 | init_Bfkmr | 94 |
| 4.27.2.6 | init_LUTs | 95 |
| 4.27.2.7 | insert_and_fetch | 95 |
| 4.27.2.8 | multiHash | 95 |
| 4.27.2.9 | read_Bfilter | 95 |
| 4.27.2.10 | save_Bfilter | 96 |
| 4.27.3 | Variable Documentation | 96 |
| 4.27.3.1 | alloc_mem | 96 |
| 4.28 | src/city.c File Reference | 96 |
| 4.28.1 | Detailed Description | 98 |
| 4.29 | src/fa_read.c File Reference | 98 |
| 4.29.1 | Detailed Description | 99 |
| 4.29.2 | Function Documentation | 99 |
| 4.29.2.1 | free_fasta | 99 |
| 4.29.2.2 | ignore_line | 99 |
| 4.29.2.3 | init_entries | 100 |
| 4.29.2.4 | init_fa | 100 |
| 4.29.2.5 | nkmers | 100 |
| 4.29.2.6 | read_fasta | 100 |
| 4.29.2.7 | realloc_fa | 101 |
| 4.29.2.8 | size_fasta | 101 |
| 4.29.2.9 | sweep_fa | 101 |
| 4.29.3 | Variable Documentation | 101 |
| 4.29.3.1 | alloc_mem | 101 |
| 4.30 | src/fopen_gen.c File Reference | 101 |
| 4.30.1 | Detailed Description | 102 |
| 4.30.2 | Function Documentation | 103 |
| 4.30.2.1 | compress | 103 |
| 4.30.2.2 | fcompress | 103 |
| 4.30.2.3 | fopen_gen | 103 |
| 4.30.2.4 | funcompress | 103 |
| 4.30.2.5 | uncompress | 103 |
| 4.31 | src/fq_read.c File Reference | 104 |
| 4.31.1 | Detailed Description | 104 |
| 4.31.2 | Function Documentation | 104 |
| 4.31.2.1 | get_fqread | 104 |

| | | |
|----------|--|-----|
| 4.31.2.2 | string_seq | 105 |
| 4.32 | src/init_makeBloom.c File Reference | 105 |
| 4.32.1 | Detailed Description | 106 |
| 4.32.2 | Variable Documentation | 106 |
| 4.32.2.1 | par_MB | 106 |
| 4.33 | src/init_makeTree.c File Reference | 106 |
| 4.33.1 | Detailed Description | 107 |
| 4.33.2 | Variable Documentation | 107 |
| 4.33.2.1 | par_MT | 107 |
| 4.34 | src/init_Qreport.c File Reference | 107 |
| 4.34.1 | Detailed Description | 108 |
| 4.34.2 | Variable Documentation | 108 |
| 4.34.2.1 | par_QR | 108 |
| 4.35 | src/init_Sreport.c File Reference | 108 |
| 4.35.1 | Detailed Description | 109 |
| 4.35.2 | Variable Documentation | 109 |
| 4.35.2.1 | par_SR | 109 |
| 4.36 | src/init_trimFilter.c File Reference | 110 |
| 4.36.1 | Detailed Description | 110 |
| 4.36.2 | Variable Documentation | 111 |
| 4.36.2.1 | par_TF | 111 |
| 4.37 | src/init_trimFilterDS.c File Reference | 111 |
| 4.37.1 | Detailed Description | 112 |
| 4.37.2 | Variable Documentation | 112 |
| 4.37.2.1 | par_TF | 112 |
| 4.38 | src/io_trimFilter.c File Reference | 112 |
| 4.38.1 | Detailed Description | 113 |
| 4.38.2 | Function Documentation | 113 |
| 4.38.2.1 | buffer_output | 113 |
| 4.39 | src/Lmer.c File Reference | 114 |
| 4.39.1 | Detailed Description | 115 |
| 4.39.2 | Function Documentation | 115 |
| 4.39.2.1 | init_map | 115 |
| 4.39.2.2 | init_map_SA | 115 |
| 4.39.3 | Variable Documentation | 115 |
| 4.39.3.1 | bw_1B | 115 |
| 4.39.3.2 | fw_1B | 115 |
| 4.39.3.3 | Nencode | 115 |
| 4.40 | src/makeBloom.c File Reference | 115 |
| 4.40.1 | Detailed Description | 116 |

| | | |
|----------|---------------------------------------|-----|
| 4.40.2 | Variable Documentation | 116 |
| 4.40.2.1 | alloc_mem | 116 |
| 4.40.2.2 | par_MB | 116 |
| 4.41 | src/makeTree.c File Reference | 117 |
| 4.41.1 | Detailed Description | 117 |
| 4.41.2 | Variable Documentation | 118 |
| 4.41.2.1 | alloc_mem | 118 |
| 4.41.2.2 | par_MT | 118 |
| 4.42 | src/Qreport.c File Reference | 118 |
| 4.42.1 | Detailed Description | 118 |
| 4.42.2 | Variable Documentation | 119 |
| 4.42.2.1 | par_QR | 119 |
| 4.43 | src/Rcommand_Qreport.c File Reference | 119 |
| 4.43.1 | Detailed Description | 120 |
| 4.43.2 | Variable Documentation | 120 |
| 4.43.2.1 | par_QR | 120 |
| 4.44 | src/Rcommand_Sreport.c File Reference | 120 |
| 4.44.1 | Detailed Description | 121 |
| 4.44.2 | Function Documentation | 121 |
| 4.44.2.1 | command_Sreport | 121 |
| 4.44.3 | Variable Documentation | 121 |
| 4.44.3.1 | par_SR | 121 |
| 4.45 | src/Sreport.c File Reference | 121 |
| 4.45.1 | Detailed Description | 122 |
| 4.45.2 | Variable Documentation | 122 |
| 4.45.2.1 | par_SR | 122 |
| 4.46 | src/stats_info.c File Reference | 122 |
| 4.46.1 | Detailed Description | 124 |
| 4.46.2 | Function Documentation | 124 |
| 4.46.2.1 | get_tile_lane | 124 |
| 4.46.2.2 | init_info | 124 |
| 4.46.2.3 | resize_info | 124 |
| 4.46.2.4 | update_ACGT_counts | 125 |
| 4.46.3 | Variable Documentation | 125 |
| 4.46.3.1 | par_QR | 125 |
| 4.47 | src/str_manip.c File Reference | 125 |
| 4.47.1 | Detailed Description | 126 |
| 4.47.2 | Macro Definition Documentation | 126 |
| 4.47.2.1 | __isascii_c | 126 |
| 4.47.3 | Function Documentation | 126 |

| | | |
|-----------|--|-----|
| 4.47.3.1 | strindex | 126 |
| 4.47.3.2 | strindexC | 126 |
| 4.47.3.3 | strsplit | 126 |
| 4.48 | src/struct_trimFilter.c File Reference | 126 |
| 4.48.1 | Detailed Description | 127 |
| 4.49 | src/tree.c File Reference | 127 |
| 4.49.1 | Detailed Description | 129 |
| 4.49.2 | Function Documentation | 129 |
| 4.49.2.1 | check_path | 129 |
| 4.49.2.2 | free_all_nodes | 129 |
| 4.49.2.3 | get_new_pool | 129 |
| 4.49.2.4 | new_node_buf | 129 |
| 4.49.2.5 | read_tree | 130 |
| 4.49.2.6 | save_tree | 130 |
| 4.49.2.7 | tree_from_fasta | 130 |
| 4.49.3 | Variable Documentation | 130 |
| 4.49.3.1 | alloc_mem | 130 |
| 4.50 | src/trim.c File Reference | 131 |
| 4.50.1 | Detailed Description | 132 |
| 4.50.2 | Macro Definition Documentation | 132 |
| 4.50.2.1 | TRIM_STRING | 132 |
| 4.50.3 | Function Documentation | 132 |
| 4.50.3.1 | align_uint32 | 132 |
| 4.50.3.2 | align_uint64 | 133 |
| 4.50.3.3 | is_read_inBloom | 134 |
| 4.50.3.4 | is_read_inTree | 134 |
| 4.50.3.5 | Nfree_Lmer | 135 |
| 4.50.3.6 | no_lowQ | 136 |
| 4.50.3.7 | no_N | 136 |
| 4.50.3.8 | Ntrim_ends | 136 |
| 4.50.3.9 | Qtrim_ends | 136 |
| 4.50.3.10 | Qtrim_endsfrac | 137 |
| 4.50.3.11 | Qtrim_frac | 137 |
| 4.50.3.12 | Qtrim_global | 137 |
| 4.50.3.13 | trim_adapter | 138 |
| 4.50.3.14 | trim_sequenceN | 138 |
| 4.50.3.15 | trim_sequenceQ | 138 |
| 4.50.4 | Variable Documentation | 139 |
| 4.50.4.1 | Nencode | 139 |
| 4.50.4.2 | par_TF | 139 |

| | | |
|----------|-----------------------------------|------------|
| 4.51 | src/trimDS.c File Reference | 139 |
| 4.51.1 | Detailed Description | 140 |
| 4.51.2 | Function Documentation | 140 |
| 4.51.2.1 | alignDS_uint64 | 140 |
| 4.51.2.2 | init_DSadap | 141 |
| 4.51.2.3 | obtain_scoreDS | 142 |
| 4.51.2.4 | pack_reads | 142 |
| 4.51.2.5 | QtrimDS | 142 |
| 4.51.2.6 | trim_adapterDS | 143 |
| 4.51.3 | Variable Documentation | 143 |
| 4.51.3.1 | bw_1B | 143 |
| 4.51.3.2 | fw_1B | 143 |
| 4.51.3.3 | par_TF | 143 |
| 4.52 | src/trimFilter.c File Reference | 143 |
| 4.52.1 | Detailed Description | 144 |
| 4.52.2 | Variable Documentation | 144 |
| 4.52.2.1 | alloc_mem | 144 |
| 4.52.2.2 | par_TF | 144 |
| 4.53 | src/trimFilterDS.c File Reference | 144 |
| 4.53.1 | Detailed Description | 145 |
| 4.53.2 | Variable Documentation | 146 |
| 4.53.2.1 | alloc_mem | 146 |
| 4.53.2.2 | par_TF | 146 |
| | Index | 147 |

Chapter 1

Class Index

1.1 Class List

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

| | | |
|------------------------------------|---|----|
| _ad_seq | Stores an adapter entry | 7 |
| _adapter | | 8 |
| _bfilter | Bloom filter structure | 9 |
| _bfkmer | Stores a processed kmer (2 bits pro nucleotide) | 10 |
| _ds_adap | Structure containing an adapter pair (for read 1 and read 2) | 11 |
| _fa_data | Stores sequences of a fasta file | 12 |
| _fa_entry | Fasta entry | 13 |
| _fq_read | Stores a fastq entry | 14 |
| _iparam_makeBloom | MakeBloom input parameters | 15 |
| _iparam_makeTree | MakeTree input parameters | 16 |
| _iparam_Qreport | Qreport input parameters | 17 |
| _iparam_Sreport | Sreport input parameters | 18 |
| _iparam_trimFilter | TrimFilter input parameters | 19 |
| _node | Node structure: formed out of T_ACGT pointers to Node structure | 22 |
| _split | Split string and the number of splitted fields | 23 |
| _stats_TF | Collects stats info from the filtering procedure | 23 |
| _stats_TFDS | Collects stats info from the filtering procedure | 24 |
| _tree | Structure containing a T_ACGT-tree | 25 |
| _uint128 | | 26 |
| statsinfo | Stores info needed to create the summary graphs | 27 |

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

| | | |
|--|---|----|
| include/ adapters.h | Sequence manipulation for alignment | 31 |
| include/ bloom.h | Functions that implement the bloom filter | 35 |
| include/ city.h | Functions for hashin strings, C translation of cityhash (C++, google) | 42 |
| include/ citycrc.h | Functions for hashin strings, C translation of cityhash (C++, google) | 43 |
| include/ defines.h | Macro definitions | 44 |
| include/ fa_read.h | Reads in and stores fasta files | 50 |
| include/ fopen_gen.h | Uncompress/compress input/output files using pipes | 53 |
| include/ fq_read.h | Fastq entries manipulations (read/write) | 54 |
| include/ init_makeBloom.h | Help dialog for makeBloom and initialization of the command line arguments | 56 |
| include/ init_makeTree.h | Help dialog for makeTree and initialization of the command line arguments | 58 |
| include/ init_Qreport.h | Header file: help dialog for Qreport and initialization of the command line arguments | 59 |
| include/ init_Sreport.h | Help dialog for Sreport and initialization of the command line arguments | 60 |
| include/ init_trimFilter.h | Help dialog for trimFilter and initialization of the command line arguments | 62 |
| include/ init_trimFilterDS.h | Help dialog for trimFilterDS and initialization of the command line arguments | 64 |
| include/ io_trimFilter.h | Buffer fq output, write summary file | 65 |
| include/ io_trimFilterDS.h | Buffer fq output, write summary file | 67 |
| include/ Lmer.h | Manipulation of Lmers and sequences | 68 |
| include/ Rcommand_Qreport.h | Get Rscript command for Qreport | 69 |
| include/ Rcommand_Sreport.h | Get Rscript command for Sreport | 70 |

| | | |
|--|---|-----|
| include/ stats_info.h | Construct the quality report variables and update them | 71 |
| include/ str_manip.h | Functions that do string manipulation | 74 |
| include/ struct_trimFilter.h | Structure where the input arguments of trimFilter and trimFilterDS will be stored and function to free the memory of it | 75 |
| include/ tree.h | Construction of tree, check paths, write tree, read in tree | 77 |
| include/ trim.h | Trims/filter sequences after Quality, N's contaminations | 82 |
| include/ trimDS.h | Trim adapters from double stranded data | 85 |
| src/ adapters.c | Sequence manipulation for alignment | 87 |
| src/ bloom.c | Functions that implement the bloom filter | 90 |
| src/ city.c | Functions for hashin strings, C translation of cityhash (C++, google) | 96 |
| src/ fa_read.c | Reads in and stores fasta files | 98 |
| src/ fopen_gen.c | Uncompress/compress input/output files using pipes | 101 |
| src/ fq_read.c | Fastq entries manipulations (read/write) | 104 |
| src/ init_makeBloom.c | Help dialog for makeBloom and initialization of the command line arguments | 105 |
| src/ init_makeTree.c | Help dialog for makeTree and initialization of the command line arguments | 106 |
| src/ init_Qreport.c | Help dialog for Qreport and initialization of the command line arguments | 107 |
| src/ init_Sreport.c | Help dialog for Sreport and initialization of the command line arguments | 108 |
| src/ init_trimFilter.c | Help dialog for trimFilter and initialization of the command line arguments | 110 |
| src/ init_trimFilterDS.c | Help dialog for trimFilterDS and initialization of the command line arguments | 111 |
| src/ io_trimFilter.c | Buffer fq output, write summary file | 112 |
| src/ Lmer.c | Manipulation of Lmers and sequences | 114 |
| src/ makeBloom.c | MakeBloom main function | 115 |
| src/ makeTree.c | MakeTree main function | 117 |
| src/ Qreport.c | QReport main function | 118 |
| src/ Rcommand_Qreport.c | Get Rscript command for Qreport | 119 |
| src/ Rcommand_Sreport.c | Get Rscript command for Sreport | 120 |
| src/ Sreport.c | Sreport main function | 121 |
| src/ stats_info.c | Construct the quality report variables and update them | 122 |
| src/ str_manip.c | Functions that do string manipulation | 125 |

| | |
|---|-----|
| src/ struct_trimFilter.c | |
| Function that frees the memory of parTF (structure storing the trimFilter/trimFilterDS input arguments) | 126 |
| src/ tree.c | |
| Construction of tree, check paths, write tree, read in tree | 127 |
| src/ trim.c | |
| Trims/filter sequences after Quality, N's contaminations | 131 |
| src/ trimDS.c | |
| Trim adapters from double stranded data | 139 |
| src/ trimFilter.c | |
| TrimFilter main function | 143 |
| src/ trimFilterDS.c | |
| TrimFilterDS main function | 144 |

Chapter 3

Class Documentation

3.1 `_ad_seq` Struct Reference

stores an adapter entry

```
#include <adapters.h>
```

Public Attributes

- int `L`
- char `seq` [READ_MAXLEN]
- int `Lpack`
- int `Lpack_sh`
- unsigned char `pack` [(READ_MAXLEN+1)/2]
- unsigned char `pack_sh` [(READ_MAXLEN+1)/2]

3.1.1 Detailed Description

stores an adapter entry

3.1.2 Member Data Documentation

3.1.2.1 `int _ad_seq::L`

length of the adapter

3.1.2.2 `int _ad_seq::Lpack`

length of the packed sequence as is

3.1.2.3 `int _ad_seq::Lpack_sh`

length of the shifted packed sequence

3.1.2.4 `unsigned char _ad_seq::pack[(READ_MAXLEN+1)/2]`

packed sequence

3.1.2.5 unsigned char _ad_seq::pack_sh[(READ_MAXLEN+1)/2]

packed shifted sequence

3.1.2.6 char _ad_seq::seq[READ_MAXLEN]

adapter sequence

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

- include/[adapters.h](#)

3.2 _adapter Struct Reference

```
#include <struct_trimFilter.h>
```

Public Attributes

- char * [ad_fa](#)
- char * [ad2_fa](#)
- int [mismatches](#)
- double [threshold](#)
- int [Nad](#)

3.2.1 Detailed Description

@ brief adapter struct @ note UNFINISHED!

3.2.2 Member Data Documentation

3.2.2.1 char* _adapter::ad2_fa

fasta file containing adapters from read 2

3.2.2.2 char* _adapter::ad_fa

fasta file containing adapters

3.2.2.3 int _adapter::mismatches

Number of allowed mismatches

3.2.2.4 int _adapter::Nad

Number of adapters

3.2.2.5 double _adapter::threshold

Score threshold

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

- include/[struct_trimFilter.h](#)

3.3 _bfilter Struct Reference

Bloom filter structure.

```
#include <bloom.h>
```

Public Attributes

- int [kmersize](#)
- int [hashNum](#)
- int [kmersizeBytes](#)
- double [falsePosRate](#)
- uint64_t [bfszBits](#)
- uint64_t [bfszBytes](#)
- uint64_t [nelem](#)
- unsigned char * [filter](#)

3.3.1 Detailed Description

Bloom filter structure.

3.3.2 Member Data Documentation

3.3.2.1 uint64_t _bfilter::bfszBits

bloom filter size (bits) (m)

3.3.2.2 uint64_t _bfilter::bfszBytes

bloom filter size (bytes)

3.3.2.3 double _bfilter::falsePosRate

False positive rate

3.3.2.4 unsigned char* _bfilter::filter

filter sequence

3.3.2.5 int _bfilter::hashNum

number of hash functions used to construct the filter

3.3.2.6 `int _bfilter::kmersize`

kmer size (number of elements)

3.3.2.7 `int _bfilter::kmersizeBytes`

Bytes needed to store the kmer (4bases ~ 1byte)

3.3.2.8 `uint64_t _bfilter::nelem`

number of elements encoded in the bloom filter (n)

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

- include/[bloom.h](#)

3.4 `_bfkmer` Struct Reference

stores a processed kmer (2 bits pro nucleotide)

```
#include <bloom.h>
```

Public Attributes

- int [kmersize](#)
- int [hashNum](#)
- int [kmersizeBytes](#)
- int [halfsizeBytes](#)
- int [hangingBases](#)
- int [hasOverhead](#)
- unsigned char * [compact](#)
- uint64_t * [hashValues](#)

3.4.1 Detailed Description

stores a processed kmer (2 bits pro nucleotide)

3.4.2 Member Data Documentation

3.4.2.1 `unsigned char* _bfkmer::compact`

encoded compactified sequence

3.4.2.2 `int _bfkmer::halfsizeBytes`

half size in bytes(needed to decide whether to store a kmer or its reverse complement)

3.4.2.3 `int _bfkmer::hangingBases`

number of hanging bases that don't complete a byte

3.4.2.4 int _bfkmer::hashNum

number of hash functions used to construct the filter

3.4.2.5 uint64_t* _bfkmer::hashValues

Values of the hash functions

3.4.2.6 int _bfkmer::hasOverhead

kmer has overhead when kmersize % 4 != 0

3.4.2.7 int _bfkmer::kmersize

kmer size (number of elements)

3.4.2.8 int _bfkmer::kmersizeBytes

Bytes needed to store the kmer (4bases ~ 1byte)

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

- [include/bloom.h](#)

3.5 _ds_adap Struct Reference

structure containing an adapter pair (for read 1 and read 2)

```
#include <trimDS.h>
```

Public Attributes

- char [ad1](#) [READ_MAXLEN]
- char [ad2](#) [READ_MAXLEN]
- int [L1](#)
- int [L2](#)

3.5.1 Detailed Description

structure containing an adapter pair (for read 1 and read 2)

3.5.2 Member Data Documentation

3.5.2.1 char _ds_adap::ad1[READ_MAXLEN]

read 1 associated adapter sequence

3.5.2.2 char _ds_adap::ad2[READ_MAXLEN]

read 2 associated adapter sequence

3.5.2.3 `int_ds_adap::L1`

adapter 1 sequence length

3.5.2.4 `int_ds_adap::L2`

adapter 2 sequence length

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

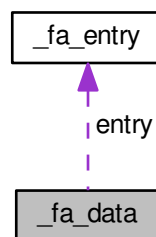
- [include/trimDS.h](#)

3.6 `_fa_data` Struct Reference

stores sequences of a fasta file

```
#include <fa_read.h>
```

Collaboration diagram for `_fa_data`:



Public Attributes

- `uint64_t` [nlines](#)
- `int` [nentries](#)
- `int` [linelen](#)
- `uint64_t` * [entrylen](#)
- [Fa_entry](#) * [entry](#)

3.6.1 Detailed Description

stores sequences of a fasta file

3.6.2 Member Data Documentation

3.6.2.1 `Fa_entry* _fa_data::entry`

Array with fasta entries (see `Fa_entry`)

3.6.2.2 `uint64_t *_fa_data::entrylen`

Array containing the length of the entries

3.6.2.3 `int _fa_data::linelen`

Line length of the `*fa` file entries

3.6.2.4 `int _fa_data::nentries`

Number of entries in `*fa` file

3.6.2.5 `uint64_t _fa_data::nlines`

Number of lines in `*fa` file

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

- `include/fa_read.h`

3.7 `_fa_entry` Struct Reference

fasta entry

```
#include <fa_read.h>
```

Public Attributes

- `uint64_t N`
- `char * seq`

3.7.1 Detailed Description

fasta entry

3.7.2 Member Data Documentation

3.7.2.1 `uint64_t _fa_entry::N`

Entry length (chars)

3.7.2.2 `char* _fa_entry::seq`

sequence

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

- `include/fa_read.h`

3.8 _fq_read Struct Reference

stores a fastq entry

```
#include <fq_read.h>
```

Public Attributes

- char [line1](#) [READ_MAXLEN]
- char [line2](#) [READ_MAXLEN]
- char [line3](#) [READ_MAXLEN]
- char [line4](#) [READ_MAXLEN]
- int [L](#)
- int [start](#)
- int [Lhalf](#)
- char [extended](#) [READ_MAXLEN]
- unsigned char [pack](#) [(READ_MAXLEN+1)/2]
- unsigned char [packsh](#) [(READ_MAXLEN+1)/2]
- int [L_ad](#)
- int [L_ext](#)
- int [L_pack](#)
- int [L_packsh](#)

3.8.1 Detailed Description

stores a fastq entry

3.8.2 Member Data Documentation

3.8.2.1 char _fq_read::extended[READ_MAXLEN]

extended sequence, adapter added to 5' end

3.8.2.2 int _fq_read::L

read length

3.8.2.3 int _fq_read::L_ad

length of adapter sequence

3.8.2.4 int _fq_read::L_ext

length of extended sequence

3.8.2.5 int _fq_read::L_pack

length of packed sequence

3.8.2.6 int _fq_read::L_packsh

length of packed sequence (shifted)

3.8.2.7 int _fq_read::Lhalf

half of read length

3.8.2.8 char _fq_read::line1[READ_MAXLEN]

Line 1 in fastq entry

3.8.2.9 char _fq_read::line2[READ_MAXLEN]

Line 2 in fastq entry

3.8.2.10 char _fq_read::line3[READ_MAXLEN]

Line 3 in fastq entry

3.8.2.11 char _fq_read::line4[READ_MAXLEN]

Line 4 in fastq entry

3.8.2.12 unsigned char _fq_read::pack[(READ_MAXLEN+1)/2]

pack sequence

3.8.2.13 unsigned char _fq_read::packsh[(READ_MAXLEN+1)/2]

pack sequence with shift

3.8.2.14 int _fq_read::start

nucleotide position start. Can only be different from zero if the read has been filtered with this tool.

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

- [include/fq_read.h](#)

3.9 _iparam_makeBloom Struct Reference

contains makeBloom input parameters

```
#include <init_makeBloom.h>
```

Public Attributes

- char * [inputfasta](#)
- char [filterfile](#) [MAX_FILENAME]
- char [paramfile](#) [MAX_FILENAME]
- int [kmersize](#)
- int [hashNum](#)
- double [falsePosRate](#)
- uint64_t [bfszBits](#)
- uint64_t [nelem](#)

3.9.1 Detailed Description

contains makeBloom input parameters

Note

nelemen will be computed once the fasta file is read and loaded.

3.9.2 Member Data Documentation

3.9.2.1 `uint64_t _iparam_makeBloom::bfszBits`

bloom filter size (bits)

3.9.2.2 `double _iparam_makeBloom::falsePosRate`

false positive rate

3.9.2.3 `char _iparam_makeBloom::filterfile[MAX_FILENAME]`

filter file path

3.9.2.4 `int _iparam_makeBloom::hashNum`

number of hash functions used to construct the filter

3.9.2.5 `char* _iparam_makeBloom::inputfasta`

fasta input file

3.9.2.6 `int _iparam_makeBloom::kmersize`

kmer size (number of elements)

3.9.2.7 `uint64_t _iparam_makeBloom::nelem`

number of elements that the bloomfilter will contain

3.9.2.8 `char _iparam_makeBloom::paramfile[MAX_FILENAME]`

param file path

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

- [include/init_makeBloom.h](#)

3.10 `_iparam_makeTree` Struct Reference

contains makeTree input parameters

```
#include <init_makeTree.h>
```

Public Attributes

- char * [inputfasta](#)
- char [outputfile](#) [MAX_FILENAME]
- int [L](#)

3.10.1 Detailed Description

contains makeTree input parameters

3.10.2 Member Data Documentation

3.10.2.1 char* _iparam_makeTree::inputfasta

fasta input file

3.10.2.2 int _iparam_makeTree::L

tree depth

3.10.2.3 char _iparam_makeTree::outputfile[MAX_FILENAME]

outputfile path

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

- include/[init_makeTree.h](#)

3.11 _iparam_Qreport Struct Reference

contains Qreport input parameters

```
#include <init_Qreport.h>
```

Public Attributes

- char * [inputfile](#)
- char [outputfilebin](#) [MAX_FILENAME]
- char [outputfilehtml](#) [MAX_FILENAME]
- char [outputfileinfo](#) [MAX_FILENAME]
- int [nQ](#)
- int [ntiles](#)
- int [minQ](#)
- int [read_len](#)
- int [filter](#)
- int [one_read_len](#)

3.11.1 Detailed Description

contains Qreport input parameters

3.11.2 Member Data Documentation

3.11.2.1 `int _iparam_Qreport::filter`

0 original data, 1 this tool filtered data, 2 other tool filtered data

3.11.2.2 `char* _iparam_Qreport::inputfile`

Inputfile name

3.11.2.3 `int _iparam_Qreport::minQ`

minimum Quality allowed 0 - 45

3.11.2.4 `int _iparam_Qreport::nQ`

different quality values (default is 46)

3.11.2.5 `int _iparam_Qreport::ntiles`

tiles (default is 96)

3.11.2.6 `int _iparam_Qreport::one_read_len`

1 all reads of equal length 0 reads have different lengths.

3.11.2.7 `char _iparam_Qreport::outputfilebin[MAX_FILENAME]`

Binary outputfile name.

3.11.2.8 `char _iparam_Qreport::outputfilehtml[MAX_FILENAME]`

html outputfile name

3.11.2.9 `char _iparam_Qreport::outputfileinfo[MAX_FILENAME]`

Info outputfile name

3.11.2.10 `int _iparam_Qreport::read_len`

original read length

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

- [include/init_Qreport.h](#)

3.12 `_iparam_Sreport` Struct Reference

contains Sreport input parameters

```
#include <init_Sreport.h>
```

Public Attributes

- char * [inputfolder](#)
- char [outputfile](#) [MAX_FILENAME]
- char * [Rmd_file](#)

3.12.1 Detailed Description

contains Sreport input parameters

3.12.2 Member Data Documentation

3.12.2.1 char* _iparam_Sreport::inputfolder

input folder

3.12.2.2 char _iparam_Sreport::outputfile[MAX_FILENAME]

html outputfile path

3.12.2.3 char* _iparam_Sreport::Rmd_file

Rmd file path

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

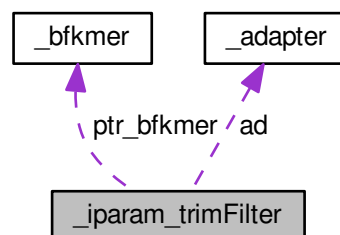
- [include/init_Sreport.h](#)

3.13 _iparam_trimFilter Struct Reference

trimFilter input parameters

```
#include <struct_trimFilter.h>
```

Collaboration diagram for _iparam_trimFilter:



Public Attributes

- char * [lfq](#)
- char * [lfq2](#)
- char * [lfa](#)
- char * [lidx](#)
- char * [linfo](#)
- char * [Oprefix](#)
- [Adapter](#) [ad](#)
- [Bfkmer](#) * [ptr_bfkmer](#)
- int [trimQ](#)
- int [trimN](#)
- int [method](#)
- bool [is_fa](#)
- bool [is_idx](#)
- bool [is_adapter](#)
- double [score](#)
- int [minQ](#)
- int [L](#)
- int [minL](#)
- int [nlowQ](#)
- int [kmersize](#)
- int [globleft](#)
- int [globright](#)
- int [percent](#)

3.13.1 Detailed Description

trimFilter input parameters

3.13.2 Member Data Documentation

3.13.2.1 [Adapter](#) [_iparam_trimFilter::ad](#)

[AdapterDS](#) trimming parameters

3.13.2.2 [int](#) [_iparam_trimFilter::globleft](#)

number of bases globally trimming from the left

3.13.2.3 [int](#) [_iparam_trimFilter::globright](#)

number of bases globally trimming from the right

3.13.2.4 [char*](#) [_iparam_trimFilter::lfa](#)

Input fa file (containing contamination sequences)

3.13.2.5 [char*](#) [_iparam_trimFilter::lfq](#)

Input fq file single stranded

3.13.2.6 char* _iparam_trimFilter::lfq2

Input fq file read 2

3.13.2.7 char* _iparam_trimFilter::lidx

Input index file (from an input.fa cont file)

3.13.2.8 char* _iparam_trimFilter::linfo

Input index info file

3.13.2.9 bool _iparam_trimFilter::is_adapter

true if filtering adapter sequences

3.13.2.10 bool _iparam_trimFilter::is_fa

true if a fasta file was passed as a parameter

3.13.2.11 bool _iparam_trimFilter::is_idx

true if an index file was passed as a parameter

3.13.2.12 int _iparam_trimFilter::kmersize

kmersize to look for contamination

3.13.2.13 int _iparam_trimFilter::L

read length

3.13.2.14 int _iparam_trimFilter::method

[TREE\(1\)](#), [SA\(2\)](#), [BLOOM\(3\)](#), 0, when not looking for cont

3.13.2.15 int _iparam_trimFilter::minL

minimum read length accepted before discarding a read

3.13.2.16 int _iparam_trimFilter::minQ

minimum quality threshold

3.13.2.17 int _iparam_trimFilter::nlowQ

maximum number of lowQ bases accepted before discarding

3.13.2.18 `char* _iparam_trimFilter::Oprefix`

Output files prefix for single str (PATH/prefix)

3.13.2.19 `int _iparam_trimFilter::percent`

percentage of lowQ bases allowed in a read

3.13.2.20 `Bfkmr* _iparam_trimFilter::ptr_bfkmr`

bloom filter kmer structure

3.13.2.21 `double _iparam_trimFilter::score`

score threshold for matching reads in sequences

3.13.2.22 `int _iparam_trimFilter::trimN`

[NO\(0\)](#), [ALL\(1\)](#), [ENDS\(2\)](#), [STRIP\(3\)](#)

3.13.2.23 `int _iparam_trimFilter::trimQ`

[NO\(0\)](#), [FRAC\(1\)](#), [ENDS\(2\)](#), [ENDSFRAC\(3\)](#), [GLOBAL\(4\)](#)

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

- [include/struct_trimFilter.h](#)

3.14 `_node` Struct Reference

Node structure: formed out of T_ACGT pointers to Node structure.

```
#include <tree.h>
```

Collaboration diagram for `_node`:



Public Attributes

- `struct _node * children [T_ACGT]`

3.14.1 Detailed Description

Node structure: formed out of T_ACGT pointers to Node structure.

3.14.2 Member Data Documentation

3.14.2.1 struct _node* _node::children[T_ACGT]

T_ACGT pointers to Node structure

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

- include/[tree.h](#)

3.15 _split Struct Reference

contains a splitted string and the number or splitted fields

```
#include <str_manip.h>
```

Public Attributes

- int [N](#)
- char ** [s](#)

3.15.1 Detailed Description

contains a splitted string and the number or splitted fields

3.15.2 Member Data Documentation

3.15.2.1 int _split::N

Number of substrings in which the string was splitted

3.15.2.2 char** _split::s

Substring array containing the splitted substrings

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

- include/[str_manip.h](#)

3.16 _stats_TF Struct Reference

collects stats info from the filtering procedure

```
#include <io_trimFilter.h>
```

Public Attributes

- int [filters](#) [NFILTERS]
- int [trimmed](#) [NFILTERS]
- int [discarded](#) [NFILTERS]
- int [good](#)
- int [nreads](#)

3.16.1 Detailed Description

collects stats info from the filtering procedure

3.16.2 Member Data Documentation

3.16.2.1 `int _stats_TF::discarded[NFILTERS]`

discarded reads: ADAP, CONT, LOWQ, NNNN

3.16.2.2 `int _stats_TF::filters[NFILTERS]`

Using filters for: ADAP, CONT, LOWQ, NNNN

3.16.2.3 `int _stats_TF::good`

good reads

3.16.2.4 `int _stats_TF::nreads`

total number of reads in the fq file

3.16.2.5 `int _stats_TF::trimmed[NFILTERS]`

trimmed reads by: ADAP, CONT, LOWQ, NNNN

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

- [include/io_trimFilter.h](#)

3.17 `_stats_TFDS` Struct Reference

collects stats info from the filtering procedure

```
#include <io_trimFilterDS.h>
```

Public Attributes

- `int` [filters](#) [`NFILTERS`]
- `int` [trimmed1](#) [`NFILTERS`]
- `int` [trimmed2](#) [`NFILTERS`]
- `int` [discarded](#) [`NFILTERS`]
- `int` [good](#)
- `int` [nreads](#)

3.17.1 Detailed Description

collects stats info from the filtering procedure

3.17.2 Member Data Documentation

3.17.2.1 int _stats_TFDS::discarded[NFILTERERS]

discarded reads: ADAP, CONT, LOWQ, NNNN

3.17.2.2 int _stats_TFDS::filters[NFILTERERS]

Using filters for: ADAP, CONT, LOWQ, NNNN

3.17.2.3 int _stats_TFDS::good

good reads

3.17.2.4 int _stats_TFDS::nreads

total number of reads in the fq file

3.17.2.5 int _stats_TFDS::trimmed1[NFILTERERS]

trimmed reads by: ADAP, CONT, LOWQ, NNNN

3.17.2.6 int _stats_TFDS::trimmed2[NFILTERERS]

trimmed reads by: ADAP, CONT, LOWQ, NNNN

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

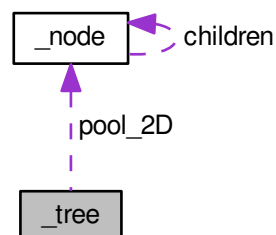
- [include/io_trimFilterDS.h](#)

3.18 _tree Struct Reference

structure containing a T_ACGT-tree.

```
#include <tree.h>
```

Collaboration diagram for _tree:



Public Attributes

- uint32_t [L](#)
- uint32_t [pool_count](#)
- uint32_t [pool_available](#)
- uint32_t [nnodes](#)
- [Node](#) ** [pool_2D](#)

3.18.1 Detailed Description

structure containing a T_ACGT-tree.

The tree structure is stored in a pointer to pointer to Node. We grow the structure on the flight as we need more memory. In the outer direction, we start by allocating NPOOL_2D pointers to Node. In the inner direction, we allocate NPOOL_1D Nodes and fill them as we read the fasta file. When all of them are allocated, we allocate again NPOOL_1D. If NPOOL_2D pointers to Node are allocated, the outer dimension is reallocated with +NPOOL_2D extra elements. L is the depth of the tree, pool_count is the number on Node* elements used so far, pool_available is the number of Nodes available in every moment, and nnodes is the total number of nodes filled in. We limit the number of allocated nodes to UINT_MAX (we cannot count more nodes!).

3.18.2 Member Data Documentation

3.18.2.1 uint32_t _tree::L

depth of the tree

3.18.2.2 uint32_t _tree::nnodes

Number of nodes in the tree

3.18.2.3 Node** _tree::pool_2D

2D pool containing the nodes that form the tree

3.18.2.4 uint32_t _tree::pool_available

Number of empty nodes available in the pool

3.18.2.5 uint32_t _tree::pool_count

Number of elements in the second dimension

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

- include/[tree.h](#)

3.19 _uint128 Struct Reference

Public Attributes

- uint64 **first**
- uint64 **second**

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

- [include/city.h](#)

3.20 statsinfo Struct Reference

stores info needed to create the summary graphs

```
#include <stats_info.h>
```

Public Attributes

- int [read_len](#)
- int [ntiles](#)
- int [nQ](#)
- int [minQ](#)
- int [tile_pos](#)
- int [nreads](#)
- int [reads_wN](#)
- int [sz_lowQ_ACGT_tile](#)
- int [sz_ACGT_tile](#)
- int [sz_reads_MlowQ](#)
- int [sz_QPosTile_table](#)
- int [sz_ACGT_pos](#)
- int * [tile_tags](#)
- int * [lane_tags](#)
- int * [qual_tags](#)
- uint64_t * [lowQ_ACGT_tile](#)
- uint64_t * [ACGT_tile](#)
- uint64_t * [reads_MlowQ](#)
- uint64_t * [QPosTile_table](#)
- uint64_t * [ACGT_pos](#)

3.20.1 Detailed Description

stores info needed to create the summary graphs

3.20.2 Member Data Documentation

3.20.2.1 uint64_t* statsinfo::ACGT_pos

A, C, G, T, N per position

3.20.2.2 uint64_t* statsinfo::ACGT_tile

A, C, G, T, N per tile, to compute the fraction of lowQuality bases per tile and per nucleotide.

3.20.2.3 int* statsinfo::lane_tags

Names of the existing tiles

3.20.2.4 uint64_t* statsinfo::lowQ_ACGT_tile

low Quality A, C, G, T, N per tile

3.20.2.5 int statsinfo::minQ

Minimum quality threshold

3.20.2.6 int statsinfo::nQ

possible quality values

3.20.2.7 int statsinfo::nreads

reads read till current position.

3.20.2.8 int statsinfo::ntiles

tiles

3.20.2.9 uint64_t* statsinfo::QPosTile_table

bases of a given quality per tile.

3.20.2.10 int* statsinfo::qual_tags

Names of the existing qualities

3.20.2.11 int statsinfo::read_len

Maximum length of a read

3.20.2.12 uint64_t* statsinfo::reads_MlowQ

reads with M(position) lowQuality bases.

3.20.2.13 int statsinfo::reads_wN

reads with N's found till current position

3.20.2.14 int statsinfo::sz_ACGT_pos

ACGT_pos size = read_len * N_ACGT

3.20.2.15 int statsinfo::sz_ACGT_tile

ACGT_tile size = ntiles * N_ACGT

3.20.2.16 int statsinfo::sz_lowQ_ACGT_tile

lowQ_ACGT_tile size = ntiles * N_ACGT

3.20.2.17 int statsinfo::sz_QPosTile_table

QposTile_Table size = ntiles * nQ * read_len

3.20.2.18 int statsinfo::sz_reads_MlowQ

reads_MlowQ size = read_len + 1

3.20.2.19 int statsinfo::tile_pos

current tile position

3.20.2.20 int* statsinfo::tile_tags

Names of the existing tiles

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

- include/[stats_info.h](#)

Chapter 4

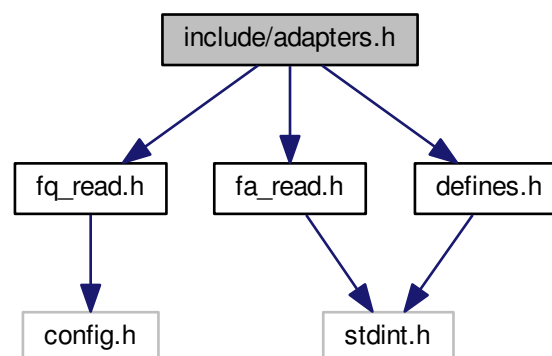
File Documentation

4.1 include/adapters.h File Reference

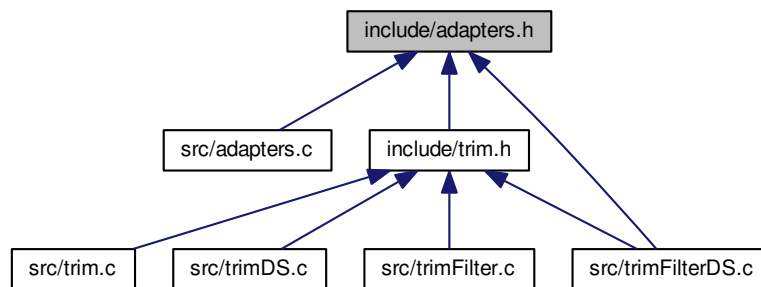
sequence manipulation for alignment

```
#include "fq_read.h"  
#include "fa_read.h"  
#include "defines.h"
```

Include dependency graph for adapters.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct `_ad_seq`
stores an adapter entry

Typedefs

- typedef struct `_ad_seq` `Ad_seq`
stores an adapter entry

Functions

- void `init_alLUTs` ()
look up table initialization for alignment (used for adapters)
- int `process_seq` (unsigned char *packed, unsigned char *read, int L, bool shift, bool isreverse)
Packs a sequence using alfw0, alfw1, albw0, albw1.
- `Ad_seq` * `pack_adapter` (`Fa_data` *ptr_fa)
*reads a **Fa_data** with adapters and stores them in an array of **Ad_seq** structs.*
- double `obtain_score` (`Fq_read` *seq, int pos_seq, `Ad_seq` *ptr_adap, int pos_ad)
computes score of a possible alignment, after having found a seed.

4.1.1 Detailed Description

sequence manipulation for alignment

Date

22.09.2017

4.1.2 Function Documentation

4.1.2.1 void init_alLUTs ()

look up table initialization for alignment (used for adapters)

It initializes: fw_1B, bw_1B. They are uint8_t arrays with 256 elements. All elements are set to 0xFF excepting the ones corresponding to 'a', 'A', 'c', 'C', 'g', 'G', 't', 'T':

| Var | a,A | c,C | g,G | t,T | Var | a,A | c,C | g,G | t,T |
|-------|------|------|------|------|-------|------|------|------|------|
| alfw0 | 0x01 | 0x02 | 0x04 | 0x08 | albw0 | 0x08 | 0x04 | 0x02 | 0x01 |
| alfw1 | 0x10 | 0x20 | 0x40 | 0x80 | albw1 | 0x80 | 0x40 | 0x20 | 0x10 |

With this variables we will encode sequences that can be compared later on. Using the bitwise XOR operator, every mismatch will amount to two bits set to 1.

4.1.2.2 double obtain_score (**Fq_read** * *seq*, int *pos_seq*, **Ad_seq** * *ptr_adap*, int *pos_ad*)

computes score of a possible alignment, after having found a seed.

The score is computed as follows:

- matching bases: score += log₁₀(4)
- unmatching bases: score -= Q/10, where Q is the quality score.

Parameters

| | |
|-----------------|--|
| <i>seq</i> | pointer to Fq_read . |
| <i>pos_seq</i> | read starting position of the alignment |
| <i>ptr_adap</i> | pointer to Ad_seq , contains the adapter info |
| <i>pos_ad</i> | adapter starting position of the alignment (reverse) |

Returns

score of the alignment

4.1.2.3 **Ad_seq*** pack_adapter (**Fa_data** * *ptr_fa*)

reads a **Fa_data** with adapters and stores them in an array of **Ad_seq** structs.

It reads the fasta structure. For every entry, an **Ad_seq** structure is allocated and the sequences are processed to create the packed sequences.

Parameters

| | |
|---------------|-------------------------------------|
| <i>ptr_fa</i> | pointer to Fa_data structure |
|---------------|-------------------------------------|

Returns

pointer to **Ad_seq**, where the information is stored.

4.1.2.4 int process_seq (unsigned char * *packed*, unsigned char * *sequence*, int *L*, bool *shift*, bool *isreverse*)

Packs a sequence using alfw0, alfw1, albw0, albw1.

It takes a sequence of length L and packs it using the look up tables into an unsigned char array, where every bytes corresponds to 2 nucleotides. One can encode the reverse complement or the sequence shifted by 1/2 byte.

Parameters

| | |
|-----------------|-------------------|
| <i>packed</i> | packed sequence |
| <i>sequence</i> | original sequence |

| | |
|------------------|---|
| <i>L</i> | original sequence length |
| <i>shift</i> | 0 if taken as is we want to shift the output 1/2 byte (>>4) |
| <i>isreverse</i> | 0 if we want the forward sequence, 1 reverse complement |

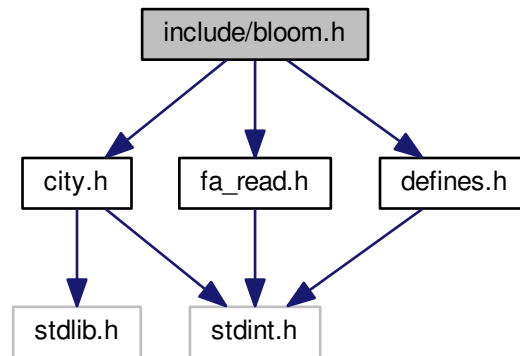
Returns

Lhalf, length in Bytes of the packed sequence

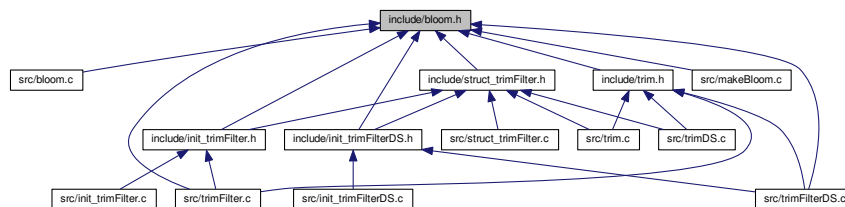
4.2 include/bloom.h File Reference

functions that implement the bloom filter

```
#include "city.h"
#include "fa_read.h"
#include "defines.h"
Include dependency graph for bloom.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- `struct _bfilter`
Bloom filter structure.
- `struct _bfkmer`
stores a processed kmer (2 bits pro nucleotide)

Typedefs

- typedef struct `_bfilter` `Bfilter`
Bloom filter structure.
- typedef struct `_bfkmer` `Bfkmer`
stores a processed kmer (2 bits pro nucleotide)

Functions

- void `init_LUTs` ()
look up table initialization
- `Bfilter * init_Bfilter` (int kmersize, uint64_t bsizeBits, int hashNum, double falsePosRate, uint64_t nelelem)
initialization of a Bfilter structure
- `Bfkmer * init_Bfkmer` (int kmersize, int hashNum)
initializes a Bfkmer structure, given the kmersize and the number of hash functions
- void `free_Bfilter` (`Bfilter *ptr_bf`)
free Bfilter memory
- void `free_Bfkmer` (`Bfkmer *ptr_bfkmer`)
free Bfkmer
- int `compact_kmer` (const unsigned char *sequence, uint64_t position, `Bfkmer *ptr_bfkmer`)
compactifies a kmer for insertion in the bloomfilter
- void `multiHash` (`Bfkmer *ptr_bfkmer`)
obtains the hashNum hashvalues for a compactified kmer
- bool `insert_and_fetch` (`Bfilter *pr_bf`, `Bfkmer *ptr_bfkmer`)
inserts the hashvalues of a kmer in filter
- bool `contains` (`Bfilter *ptr_bf`, `Bfkmer *ptr_bfkmer`)
check if kmer is contained in the filter
- `Bfilter * create_Bfilter` (`Fa_data *ptr_fasta`, int kmersize, uint64_t bsizeBits, int hashNum, double falsePosRate, uint64_t nelelem)
creates a bloom filter from a fasta structure.
- void `save_Bfilter` (`Bfilter *ptr_bf`, char *filterfile, char *paramfile)
saves a bloomfilter to disk
- `Bfilter * read_Bfilter` (char *filterfile, char *paramfile)
reads a bloom filter from a file

Variables

- static const unsigned char `bitMask` [0x08]
bitMask, ith bit set to 1 in position i

4.2.1 Detailed Description

functions that implement the bloom filter

Author

Paula Perez paulaperezrubio@gmail.com

Date

04.09.2017

4.2.2 Function Documentation

4.2.2.1 `int compact_kmer (const unsigned char * sequence, uint64_t position, Bfkmer * ptr_bfkmer)`

compactifies a kmer for insertion in the bloomfilter

Parameters

| | |
|-------------------|--|
| <i>sequence</i> | unsigned char DNA sequence (or cDNA) |
| <i>position</i> | position in the sequence where the kmer starts |
| <i>ptr_bfkmer</i> | initialized Bfkmer |

The compactified sequence is computed in the following way:

- We start compactifying both, the forward and backward (reverse complement). The outer loop covers up until half of the sequence.
- As soon as one of the two is lexicographically smaller, we continue only with it. In that way, the "smaller" sequence is consistently returned.
- If the sequence is palindromic, we continue with the forward sequence.
- kmer size should be > 3 .

We illustrate the compactification with an example:

```
kmer = TTTT|GGAT
m_fw = 00000000 | 00000000 // 2 bytes
m_bw = 00000000 | 00000000 // 2 bytes
m_fw[0] |= fw0['T'] = 0xC0|0x00; m_bw[0] |= bw0['T'] = 0x00|0x00;
m_fw[0] |= fw1['T'] = 0xF0|0x00; m_bw[0] |= bw1['A'] = 0x30|0x00;
m_fw[0] |= fw2['T'] = 0xFC|0x00; m_bw[0] |= bw2['G'] = 0x34|0x00;
m_fw[0] |= fw3['T'] = 0xFF|0x00; m_bw[0] |= bw3['G'] = 0x35|0x00;
m_fw[1] |= fw0['G'] = 0xC0|0x80; m_bw[1] |= bw0['T'] = 0x35|0x00;
m_fw[1] |= fw1['G'] = 0xF0|0xA0; m_bw[1] |= bw1['T'] = 0x35|0x00;
m_fw[1] |= fw2['A'] = 0xFC|0xA0; m_bw[1] |= bw2['T'] = 0x35|0x00;
m_fw[1] |= fw3['T'] = 0xFF|0xA3; m_bw[1] |= bw3['T'] = 0x35|0x00;
```

(In this case, we would store m_bw)

4.2.2.2 bool contains (Bfilter * ptr_bf, Bfkmer * ptr_bfkmer)

check if kmer is contained in the filter

Parameters

| | |
|-------------------|---|
| <i>ptr_bf</i> | pointer to a Bfilter structure, where a bloomfilter is stored |
| <i>ptr_bfkmer</i> | pointer to a Bfkmer structure containing the hash values |

Returns

true if all corresponding bits were set to 1 in the filter

4.2.2.3 Bfilter* create_Bfilter (Fa_data * ptr_fasta, int kmer size, uint64_t bsizeBits, int hashNum, double falsePosRate, uint64_t nelelem)

creates a bloom filter from a fasta structure.

Parameters

| | |
|------------------|--|
| <i>ptr_fasta</i> | pointer to fasta structure |
| <i>kmer size</i> | length of kmers to be inserted in the filter |
| <i>bsizeBits</i> | size of Bloom filter in bits |
| <i>hashNum</i> | number of hash functions to be used |

| | |
|---------------------|--|
| <i>falsePosRate</i> | false positive rate |
| <i>nelem</i> | number of elemens (kmers in the sequece) contained in the filter |

Returns

pointer to Bloom filter structure, where the fasta file was encoded.

4.2.2.4 Bfilter* init_Bfilter (int *kmerSize*, uint64_t *bfsSizeBits*, int *hashNum*, double *falsePosRate*, uint64_t *nelem*)

initialization of a Bfilter structure

Parameters

| | |
|---------------------|--|
| <i>kmerSize</i> | number of elements of the kmer |
| <i>bfsSizeBits</i> | size of the bloomfilter (in Bits) |
| <i>hashNum</i> | number of hash functions to be computed |
| <i>falsePosRate</i> | false positive rate |
| <i>nelem</i> | number of elemens (kmers in the sequece) contained in the filter |

Returns

pointer to initialized Bfilter structure

Given a *kmerSize*, *bfsSizeBits*, number of hash functions, we assign these values to the struture and the two additional values: *kmerSizeBytes* = (*kmerSize* + BASESINCHAR - 1)/BASESINCHAR

4.2.2.5 Bfkmer* init_Bfkmer (int *kmerSize*, int *hashNum*)

initializes a Bfkmer structure, given the *kmerSize* and the number of hash functions

Parameters

| | |
|-----------------|---|
| <i>kmerSize</i> | number of elements of the kmer |
| <i>hashNum</i> | number of hash functions to be computed |

Returns

pointer to a Bfkmer structure

kmerSizeBytes, *halfSizeBytes*, *hangingBases*, *hasOverhead* *hashNum* are assigned and memory is allocated and set to 0 for compact and *hashValues*

4.2.2.6 void init_LUTs ()

look up table initialization

It initializes: *fw0*, *fw1*, *fw2*, *fw3*, *bw0*, *bw2*, *bw3*, *bw4*. They are uint8_t arrays with 256 elements. All elements are set to 0xFF excepting the ones corresponding to 'a', 'A', 'c', 'C', 'g', 'G', 't', 'T':

| Var | a,A | c,C | g,G | t,T | Var | a,A | c,C | g,G | t,T |
|-----|------|------|------|------|-----|------|------|------|------|
| fw0 | 0x00 | 0x40 | 0x80 | 0xC0 | bw0 | 0xC0 | 0x80 | 0x40 | 0x00 |
| fw1 | 0x00 | 0x10 | 0x20 | 0x30 | bw1 | 0x30 | 0x20 | 0x10 | 0x00 |
| fw2 | 0x00 | 0x04 | 0x08 | 0x0C | bw2 | 0x0C | 0x08 | 0x04 | 0x00 |
| fw3 | 0x00 | 0x01 | 0x02 | 0x03 | bw3 | 0x03 | 0x02 | 0x01 | 0x00 |

With these variables, we will be able to encode a Sequence using 2 bits per nucleotide.

4.2.2.7 **bool** insert_and_fetch (**Bfilter** * *ptr_bf*, **Bfkmer** * *ptr_bfkmer*)

inserts the hashvalues of a kmer in filter

Parameters

| | |
|-------------------|---|
| <i>ptr_bf</i> | pointer to Bfilter structure, where we will include the new entry |
| <i>ptr_bfkmer</i> | pointer to Bfkmer structure, where the hashvalues are stored |

Returns

true if the positions of the hash values were already set to one previously.

The hash values are inserted in the following way.

- modValue = hashvalue mod(filter size) is calculated.
- the bit in position modValue of the filter is set to 1.

4.2.2.8 void multiHash (Bfkmer * *ptr_bfkmer*)

obtains the hashNum hashvalues for a compactified kmer

The hash values are computed using the CityHash64 hash functions.

4.2.2.9 Bfilter* read_Bfilter (char * *filterfile*, char * *paramfile*)

reads a bloom filter from a file

Parameters

| | |
|-------------------|------------------------------------|
| <i>filterfile</i> | path to file containing the filter |
| <i>paramfile</i> | path to file containing the filter |

Returns

a pointer to a filter structure containing the bloomfilter

This function reads two files, the auxiliar inputfile where kmersize, hashNum and bsizeBits are stored, and the actual filter file. If one of them is missing, the program exits with an error. If successful, a pointer to a Bfilter structure with the bloom filter is return

4.2.2.10 void save_Bfilter (Bfilter * *ptr_bf*, char * *filterfile*, char * *paramfile*)

saves a bloomfilter to disk

Parameters

| | |
|-------------------|--|
| <i>ptr_bf</i> | pointer to Bfilter structure (contains the filter) |
| <i>filterfile</i> | path to file where the output will be stored |
| <i>paramfile</i> | path to file where the prameters will be stored |

This function will save the bloomfilter in the path filterfile. The paramfile will store the following data:

- kmersize
- hashNum
- bsizeBits
- falsePosRate
- nelem

4.2.3 Variable Documentation

4.2.3.1 `const unsigned char bitMask[0x08]` `[static]`

Initial value:

```
= {0x01, 0x02, 0x04, 0x08,
    0x10, 0x20, 0x40, 0x80}
```

`bitMask`, `ith` bit set to 1 in position `i`

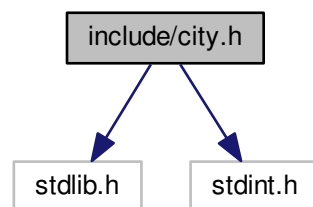
4.3 `include/city.h` File Reference

functions for hashin strings, C translation of cityhash (C++, google)

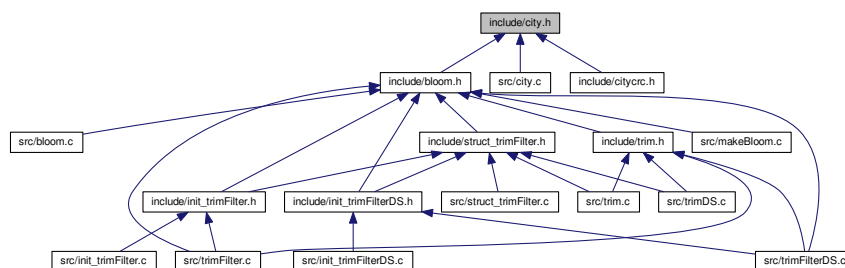
```
#include <stdlib.h>
```

```
#include <stdint.h>
```

Include dependency graph for `city.h`:



This graph shows which files directly or indirectly include this file:



Classes

- `struct uint128`

Macros

- `#define UInt128Low64(x) (x).first`
- `#define UInt128High64(x) (x).second`

Typedefs

- typedef uint8_t **uint8**
- typedef uint16_t **uint16**
- typedef uint32_t **uint32**
- typedef uint64_t **uint64**
- typedef struct _uint128 **uint128**

Functions

- uint64_t **CityHash64** (const char *buf, size_t len)
- uint64_t **CityHash64WithSeed** (const char *buf, size_t len, uint64_t seed)
- uint64_t **CityHash64WithSeeds** (const char *buf, size_t len, uint64_t seed0, uint64_t seed1)
- **uint128** **CityHash128** (const char *s, size_t len)
- **uint128** **CityHash128WithSeed** (const char *s, size_t len, **uint128** seed)
- uint32 **CityHash32** (const char *buf, size_t len)
- static uint64_t **Hash128to64** (const **uint128** x)

4.3.1 Detailed Description

functions for hashin strings, C translation of cityhash (C++, google)

Author

bdnt

See also

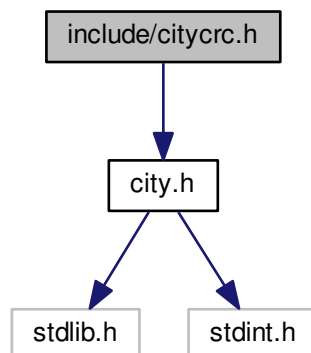
<https://github.com/bdnt/cityhash-c>
<https://github.com/google/cityhash>

4.4 include/citycrc.h File Reference

functions for hashin strings, C translation of cityhash (C++, google)

```
#include "city.h"
```

Include dependency graph for citycrc.h:



Functions

- [uint128](#) **CityHashCrc128** (const char *s, size_t len)
- [uint128](#) **CityHashCrc128WithSeed** (const char *s, size_t len, [uint128](#) seed)
- void **CityHashCrc256** (const char *s, size_t len, uint64 *result)

4.4.1 Detailed Description

functions for hashin strings, C translation of cityhash (C++, google)

Author

bdnt

See also

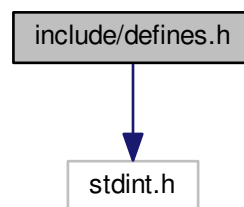
<https://github.com/bdnt/cityhash-c>
<https://github.com/google/cityhash>

4.5 include/defines.h File Reference

Macro definitions.

```
#include <stdint.h>
```

Include dependency graph for defines.h:



This graph shows which files directly or indirectly include this file:



Macros

- [#define](#) **B_LEN** 131072
- [#define](#) **MAX_FILENAME** 300
- [#define](#) **bool** int16_t
- [#define](#) **true** 1

- `#define false 0`
- `#define max(a, b) (((a) > (b)) ? (a) : (b))`
- `#define min(a, b) (((a) < (b)) ? (a) : (b))`
- `#define mem_usageMB()`
- `#define mem_usage()`
- `#define DEFAULT_MINQ 27`
- `#define DEFAULT_NTILES 96`
- `#define DEFAULT_NQ 46`
- `#define ZEROQ 33`
- `#define N_ACGT 5`
- `#define MAX_RCOMMAND 4000`
- `#define FA_ENTRY_BUF 20`
- `#define LOG_4 0.60206`
- `#define MIN_NMATCHES 12`
- `#define T_ACGT 4`
- `#define NPOOL_1D 1048576`
- `#define NPOOL_2D 16`
- `#define MAX_FASZ_TREE 1e7`
- `#define BITSPERCHAR 8`
- `#define BASESPERCHAR 4`
- `#define KMER_LEN 25`
- `#define FALSE_POS_RATE 0.05`
- `#define ZERO_POS_RATE 1e-14`
- `#define NO 0`
- `#define ALL 1`
- `#define ENDS 2`
- `#define STRIP 3`
- `#define FRAC 3`
- `#define ENDSFRAC 4`
- `#define GLOBAL 5`
- `#define TREE 1`
- `#define SA 2`
- `#define BLOOM 3`
- `#define ERROR 1000`
- `#define DEFAULT_MINL 25`
- `#define ADAP 0`
- `#define CONT 1`
- `#define LOWQ 2`
- `#define NNNN 3`
- `#define GOOD 4`
- `#define NFILTERS 4`
- `#define ADAP2 5`
- `#define CONT2 6`
- `#define LOWQ2 7`
- `#define NNNN2 8`
- `#define GOOD2 9`
- `#define NFILES_DS 10`

4.5.1 Detailed Description

Macro definitions.

Author

Paula Perez paulaperezrubio@gmail.com

Date

07.08.2017

4.5.2 Macro Definition Documentation

4.5.2.1 `#define ADAP 0`

Adapter filter

4.5.2.2 `#define ADAP2 5`

Adapter filter read2

4.5.2.3 `#define ALL 1`

Trims if a lowQ base calling | N is found

4.5.2.4 `#define B_LEN 131072`

buffer size

4.5.2.5 `#define BASESPERCHAR 4`

number of nucleotides that can fit in a char

4.5.2.6 `#define BITSPERCHAR 8`

number of bits in a char

4.5.2.7 `#define BLOOM 3`

Use a bloom filter to look for contaminations

4.5.2.8 `#define bool int16_t`

define a bool type

4.5.2.9 `#define CONT 1`

Contamination filter

4.5.2.10 #define CONT2 6

Contamination filter read2

4.5.2.11 #define DEFAULT_MINL 25

Default minimum length under which we discard the reads

4.5.2.12 #define DEFAULT_MINQ 27

Minimum quality threshold

4.5.2.13 #define DEFAULT_NQ 46

Default number of different quality values

4.5.2.14 #define DEFAULT_NTILES 96

Default number of tiles

4.5.2.15 #define ENDS 2

Trims at the ends

4.5.2.16 #define ENDSFRAC 4

trims at the ends and discards a read if the remaining part has more than > percent lowQ bases

4.5.2.17 #define ERROR 1000

Encodes an error when reading in trimN, trimQ, method options in trimFilter

4.5.2.18 #define FA_ENTRY_BUF 20

buffer for fasta entries

4.5.2.19 #define false 0

assign false to 0

4.5.2.20 #define FALSE_POS_RATE 0.05

default false positive rate

4.5.2.21 #define FRAC 3

Discards a read if it contains > percent lowQ bases

4.5.2.22 `#define GLOBAL 5`

Trims a fixed # bases from e left and right

4.5.2.23 `#define GOOD 4`

Good reads

4.5.2.24 `#define GOOD2 9`

Good reads read2

4.5.2.25 `#define KMER_LEN 25`

default kmer length

4.5.2.26 `#define LOG_4 0.60206`

$\log_{10}(4)$ for the adapters alignment score

4.5.2.27 `#define LOWQ 2`

Low quality filter

4.5.2.28 `#define LOWQ2 7`

Low quality filter read2

4.5.2.29 `#define max(a, b) (((a) > (b)) ? (a) : (b))`

max function

4.5.2.30 `#define MAX_FASZ_TREE 1e7`

Maximum fasta size for constructing a tree. DECIDE A SENSIBLE SIZE

4.5.2.31 `#define MAX_FILENAME 300`

Maximum # chars in a filename

4.5.2.32 `#define MAX_RCOMMAND 4000`

Maximum # chars in R command

4.5.2.33 `#define mem_usage()`

Value:

```
fprintf(stderr, \
    "- Current allocated memory: %ld Bytes.\n", \
    alloc_mem)
```

returns allocated memory in Bytes

4.5.2.34 #define mem_usageMB()

Value:

```
fprintf(stderr, \
    "- Current allocated memory: %ld MB.\n", \
    alloc_mem >> 20)
```

returns allocated memory in MB

4.5.2.35 #define min(a, b) ((a) < (b)) ? (a) : (b))

min function

4.5.2.36 #define MIN_NMATCHES 12

minimum number of matches demanded

4.5.2.37 #define N_ACGT 5

Number of different nucleotides in the fq file

4.5.2.38 #define NFILES_DS 10

number of outputfiles in double stranded case

4.5.2.39 #define NFILTERS 4

total number of filters

4.5.2.40 #define NNNN 3

N's presence filter

4.5.2.41 #define NNNN2 8

N's presence filter read2

4.5.2.42 #define NO 0

No trimming

4.5.2.43 #define NPOOL_1D 1048576

Number of Node structs allocated in inner dim

4.5.2.44 #define NPOOL_2D 16

Number of *Node allocated in outer dim

4.5.2.45 `#define SA 2`

Use a suffix array to look for contaminations

4.5.2.46 `#define STRIP 3`

Looks for the largest N-free sequence

4.5.2.47 `#define T_ACGT 4`

Number of children per node in tree

4.5.2.48 `#define TREE 1`

Use a tree to look for contaminations

4.5.2.49 `#define true 1`

assign true to 1

4.5.2.50 `#define ZERO_POS_RATE 1e-14`

0 threshold for a double

4.5.2.51 `#define ZEROQ 33`

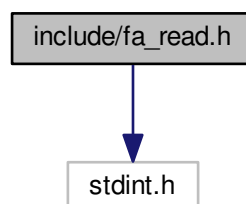
ASCII code of lowest quality value (!)

4.6 `include/fa_read.h` File Reference

reads in and stores fasta files

```
#include <stdint.h>
```

Include dependency graph for `fa_read.h`:



Parameters

| | |
|---------------|-------------------------------|
| <i>ptr_fa</i> | pointer to Fa_data structure. |
|---------------|-------------------------------|

The dynamically allocated memory in a Fa_data struct is deallocated and counted, so that we can

4.6.2.2 uint64_t nkmers (Fa_data * ptr_fa, int kmerize)

number of kmers of length kmerize contained in a fasta structure

Returns

number of kmers of length kmerize contained in a fasta structure

4.6.2.3 int read_fasta (char * filename, Fa_data * ptr_fa)

reads a fasta file and stores the contents in a Fa_data structure.

Parameters

| | |
|-----------------|-------------------------------|
| <i>filename</i> | path to a fasta input file. |
| <i>ptr_fa</i> | pointer to Fa_data structure. |

Returns

number of entries in the fasta file.

A fasta file is read and stored in a structure Fa_data. The basic problem with reading FASTA files is that there is no end-of-record indicator. When you're reading sequence n, you don't know you're done until you've read the header line for sequence n+1, which you won't parse 'til later (when you're reading in the sequence n+1). The solution implemented here is to read the file twice. The first time, (sweep_fa), we initialize Fa_data and store the parameters:

- nlines: number of lines of the fasta file.
- nentries: number of entries in the fasta file.
- linelen: length of a line in the considered fasta file.
- entrylen: array containing the lengths of every entry. With this information, the pointer to Fa_entry can be allocated and the file is read again and the entries are stored in the structure.

4.6.2.4 uint64_t size_fasta (Fa_data * ptr_fa)

computes length of genome in fasta structure

Parameters

| | |
|---------------|--------------------|
| <i>ptr_fa</i> | pointer to Fa_data |
|---------------|--------------------|

Returns

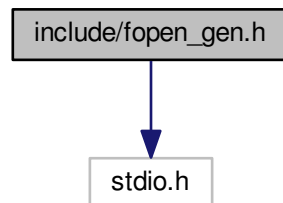
total number of nucleotides

4.7 include/fopen_gen.h File Reference

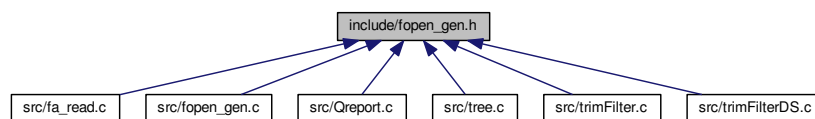
Uncompress/compress input/output files using pipes.

```
#include <stdio.h>
```

Include dependency graph for fopen_gen.h:



This graph shows which files directly or indirectly include this file:



Macros

- `#define READ_END 0`
- `#define WRITE_END 1`
- `#define PERMISSIONS 0640`

Functions

- `int setCloexec (int fd)`
- `FILE * fopen_gen (const char *path, const char *mode)`

Generalized fopen function. fopen_gen is to be used as fopen. Can be used in read and in write mode. When used in read mode with a compressed extension, the file will be first decompressed and then read. When used in write mode with a compressed extension, the output will be compressed.

4.7.1 Detailed Description

Uncompress/compress input/output files using pipes.

Hook the standard file opening functions, open, fopen and fopen64. If the extension of the file being opened indicates the file is compressed (.gz, .bz2, .xz), when opening in the reading mode a pipe to a program is opened that decompresses that file (gunzip, bunzip2 or xzdec) and return a handle to the open pipe. When opening in the writing mode (only for .gz, .bam), a pipe to a program is opened that compresses the output.

Author

Paula Perez paulaperezrubio@gmail.com

Date

03.08.2017

Warning

vfork vs fork to be checked!

Note

- original copyright note - (reading mode, original C++ code) author: Shaun Jackman sjackman@bcgsc.ca, <https://github.com/bcgsc>, filename: Uncompress.cpp

4.7.2 Function Documentation

4.7.2.1 FILE* fopen_gen (const char * *path*, const char * *mode*)

Generalized fopen function. fopen_gen is to be used as fopen. Can be used in read and in write mode. When used in read mode with a compressed extension, the file will be first decompressed and then read. When used in write mode with a compressed extension, the output will be compressed.

Returns

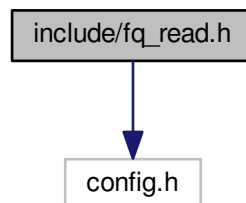
a FILE pointer

4.8 include/fq_read.h File Reference

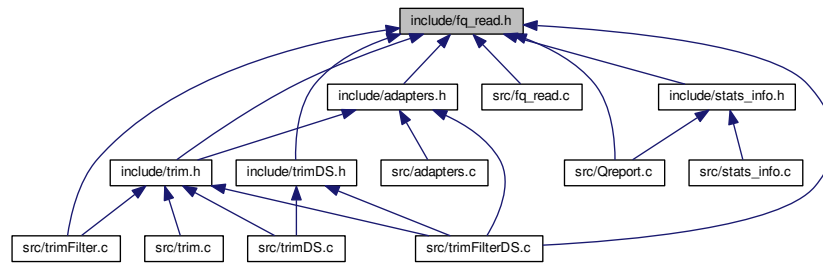
fastq entries manipulations (read/write)

```
#include "config.h"
```

Include dependency graph for fq_read.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [_fq_read](#)
stores a fastq entry

Typedefs

- typedef struct [_fq_read](#) [Fq_read](#)
stores a fastq entry

Functions

- int [get_fqread](#) ([Fq_read](#) *seq, char *buffer, int pos1, int pos2, int nline, int read_len, int filter)
reads fastq line from a buffer
- int [string_seq](#) ([Fq_read](#) *seq, char *char_seq)
writes the fq entry in a string

4.8.1 Detailed Description

fastq entries manipulations (read/write)

Author

Paula Perez paulaperezrubio@gmail.com

Date

03.08.2017

4.8.2 Function Documentation

4.8.2.1 int get_fqread ([Fq_read](#) * seq, char * buffer, int pos1, int pos2, int nline, int read_len, int filter)

reads fastq line from a buffer

a fastq line is read from a buffer and the relevant information is stored in a structure **Fq_read**. Depending on the value of **filter**, information about whether the read was trimmed is stored.

Parameters

| | |
|-----------------|--|
| <i>seq</i> | pointer to Fq_read , where the info will be stored. |
| <i>buffer</i> | variable where the file being read is stored. |
| <i>pos1</i> | buffer start position of the line. |
| <i>pos2</i> | buffer end position of the line. |
| <i>nline</i> | file line number being read. |
| <i>read_len</i> | predefined read length |
| <i>filter</i> | 0 original file, 1 file filtered with filter_trim, 2 file filtered with another tool |

4.8.2.2 int string_seq (Fq_read * seq, char * char_seq)

writes the fq entry in a string

Parameters

| | |
|-----------------|--|
| <i>seq</i> | pointer to Fq_read , where the info will be stored. |
| <i>char_seq</i> | pointer to buffer, where the sequence will be stored |

Warning

change the call to sprintf to snprintf

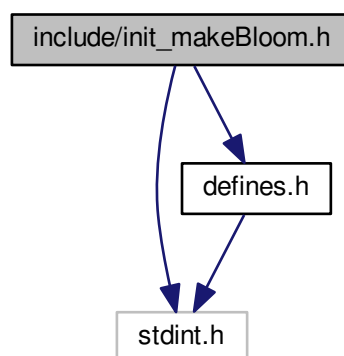
4.9 include/init_makeBloom.h File Reference

Help dialog for makeBloom and initialization of the command line arguments.

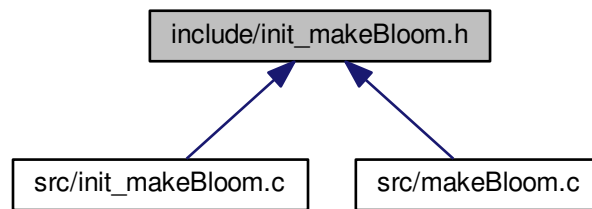
```
#include <stdint.h>
```

```
#include "defines.h"
```

Include dependency graph for init_makeBloom.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [_iparam_makeBloom](#)
contains makeBloom input parameters

Typedefs

- typedef struct [_iparam_makeBloom](#) [lparam_makeBloom](#)
contains makeBloom input parameters

Functions

- void [printHelpDialog_makeBloom](#) ()
Function that prints makeBloom help dialog when called.
- void [getarg_makeBloom](#) (int argc, char **argv)
Reads in the arguments passed through the command line to makeBloom. and stores them in the global variable par_MB.

4.9.1 Detailed Description

Help dialog for makeBloom and initialization of the command line arguments.

Author

Paula Perez paulaperezrubio@gmail.com

Date

05.09.2017

4.9.2 Typedef Documentation

4.9.2.1 typedef struct [_iparam_makeBloom](#) [lparam_makeBloom](#)

contains makeBloom input parameters

Note

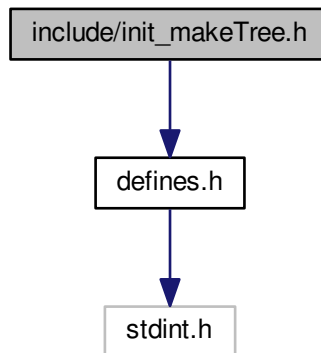
nelemen will be computed once the fasta file is read and loaded.

4.10 include/init_makeTree.h File Reference

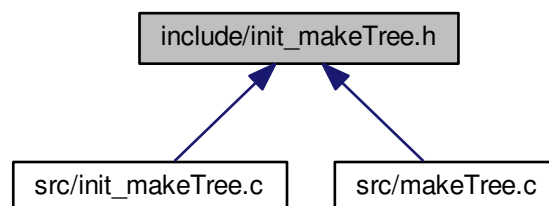
Help dialog for makeTree and initialization of the command line arguments.

```
#include "defines.h"
```

Include dependency graph for init_makeTree.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [_iparam_makeTree](#)
contains makeTree input parameters

Typedefs

- typedef struct [_iparam_makeTree](#) [lparam_makeTree](#)

contains makeTree input parameters

Functions

- void [printHelpDialog_makeTree](#) ()

Function that prints makeTree help dialog when called.

- void [getarg_makeTree](#) (int argc, char **argv)

Reads in the arguments passed through the command line to makeTree. and stores them in the global variable par_MT.

4.10.1 Detailed Description

Help dialog for makeTree and initialization of the command line arguments.

Author

Paula Perez paulaperezrubio@gmail.com

Date

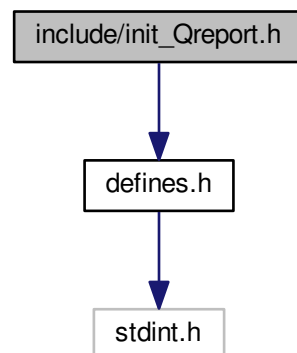
23.08.2017

4.11 include/init_Qreport.h File Reference

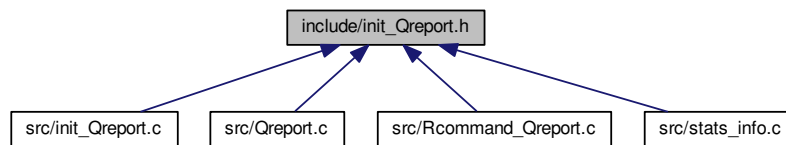
Header file: help dialog for Qreport and initialization of the command line arguments.

#include "defines.h"

Include dependency graph for init_Qreport.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [_iparam_Qreport](#)
contains Qreport input parameters

Typedefs

- typedef struct [_iparam_Qreport](#) [Iparam_Qreport](#)
contains Qreport input parameters

Functions

- void [printHelpDialog_Qreport](#) ()
Function that prints Qreport help dialog when called.
- void [getarg_Qreport](#) (int argc, char **argv)
Reads in the arguments passed through the command line to Qreport. and stores them in the global variable par_QR.

4.11.1 Detailed Description

Header file: help dialog for Qreport and initialization of the command line arguments.

Author

Paula Perez paulaperezrubio@gmail.com

Date

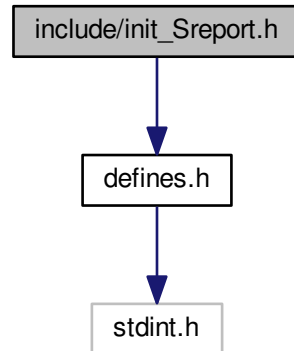
03.08.2017

4.12 include/init_Sreport.h File Reference

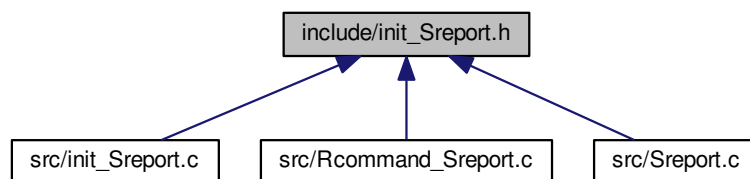
Help dialog for Sreport and initialization of the command line arguments.

```
#include "defines.h"
```

Include dependency graph for init_Sreport.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [_iparam_Sreport](#)
contains Sreport input parameters

Typedefs

- typedef struct [_iparam_Sreport](#) [lparam_Sreport](#)
contains Sreport input parameters

Functions

- void [printHelpDialog_Sreport](#) ()
Function that prints Sreport help dialog when called.
- void [getarg_Sreport](#) (int argc, char **argv)
Reads in the arguments passed through the command line to Sreport. and stores them in the global variable par_SR.

4.12.1 Detailed Description

Help dialog for Sreport and initialization of the command line arguments.

Author

Paula Perez paulaperezrubio@gmail.com

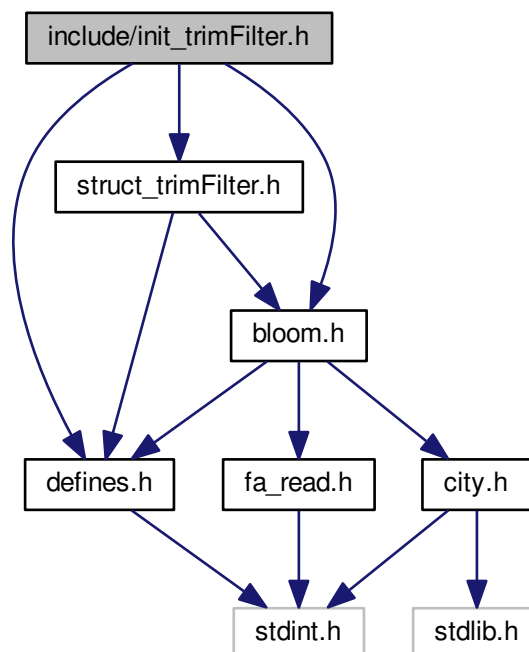
Date

09.08.2017

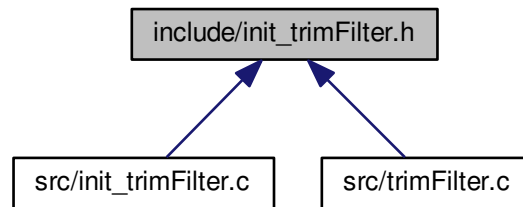
4.13 include/init_trimFilter.h File Reference

help dialog for trimFilter and initialization of the command line arguments.

```
#include "defines.h"
#include "bloom.h"
#include "struct_trimFilter.h"
Include dependency graph for init_trimFilter.h:
```



This graph shows which files directly or indirectly include this file:



Functions

- void [printHelpDialog_trimFilter](#) ()

Function that prints trimFilter help dialog when called.

- void [getarg_trimFilter](#) (int argc, char **argv)

Reads in the arguments passed through the command line to trimFilter. and stores them in the global variable par_TF.

4.13.1 Detailed Description

help dialog for trimFilter and initialization of the command line arguments.

Author

Paula Perez paulaperezrubio@gmail.com

Date

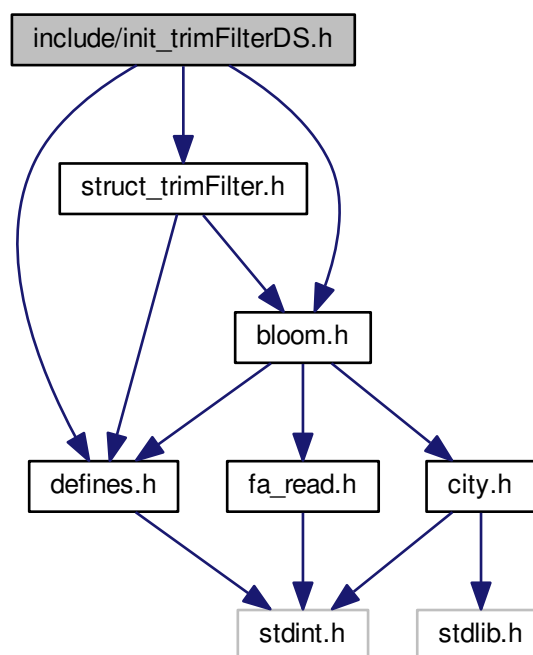
24.08.2017

4.14 include/init_trimFilterDS.h File Reference

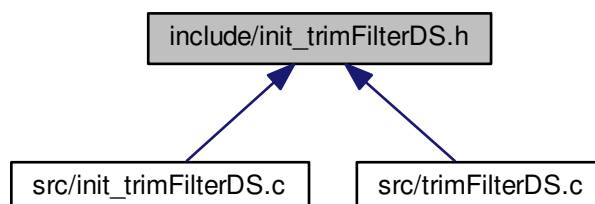
help dialog for trimFilterDS and initialization of the command line arguments.

```
#include "defines.h"  
#include "bloom.h"  
#include "struct_trimFilter.h"
```

Include dependency graph for init_trimFilterDS.h:



This graph shows which files directly or indirectly include this file:



Functions

- void [printHelpDialog_trimFilterDS](#) ()

Function that prints trimFilterDS help dialog when called.

- void [getarg_trimFilterDS](#) (int argc, char **argv)

Reads in the arguments passed through the command line to trimFilterDS and stores them in the global variable `par_TF`.

4.14.1 Detailed Description

help dialog for trimFilterDS and initialization of the command line arguments.

Author

Paula Perez paulaperezrubio@gmail.com

Date

07.10.2017

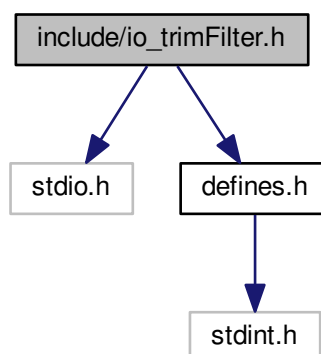
4.15 include/io_trimFilter.h File Reference

buffer fq output, write summary file

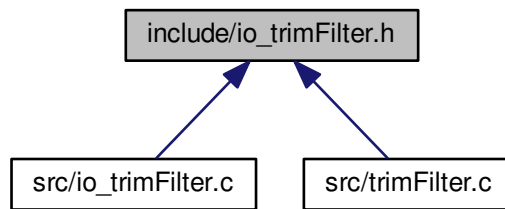
```
#include <stdio.h>
```

```
#include "defines.h"
```

Include dependency graph for io_trimFilter.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct `_stats_TF`
collects stats info from the filtering procedure

Typedefs

- typedef struct `_stats_TF` `Stats_TF`
collects stats info from the filtering procedure

Functions

- void `buffer_output` (FILE *fout, const char *a, const int len, const int fd_i)
buffers the output before writing to disk, writes out summary
- void `write_summary_TF` (Stats_TF tf_stats, char *filename)
writes stats of filtering to summary file (binary)

4.15.1 Detailed Description

buffer fq output, write summary file

Author

Paula Perez paulaperezrubio@gmail.com

Date

29.08.2017

4.15.2 Function Documentation

4.15.2.1 void `buffer_output` (FILE * *fout*, const char * *str*, const int *len*, const int *fd_i*)

buffers the output before writing to disk, writes out summary

Parameters

| | |
|-------------|--|
| <i>fout</i> | FILE pointer where we might write to disk; |
| <i>str</i> | string we want to add |
| <i>len</i> | length of the string we want to add |
| <i>fd_i</i> | identifier: GOOD, ADAP, CONT, LOWQ, NNNN |

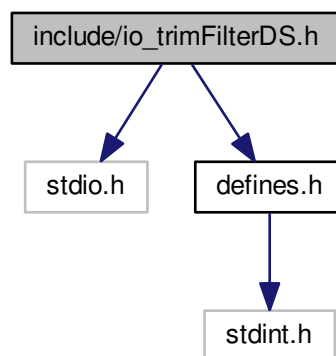
4.16 include/io_trimFilterDS.h File Reference

buffer fq output, write summary file

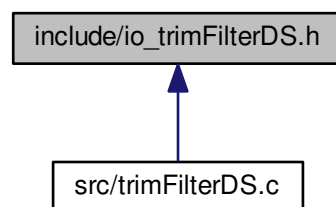
```
#include <stdio.h>
```

```
#include "defines.h"
```

Include dependency graph for io_trimFilterDS.h:



This graph shows which files directly or indirectly include this file:



Classes

- [struct `_stats_TFDS`](#)

collects stats info from the filtering procedure

Typedefs

- typedef struct [_stats_TFDS](#) Stats_TFDS
collects stats info from the filtering procedure

Functions

- void [buffer_outputDS](#) (FILE *fout, const char *a, const int len, const int fd_i)
buffers the output before writing to disk
- void [write_summary_TFDS](#) (Stats_TFDS tfds_stats, char *filename)
writes stats of filtering to summary file (binary)

4.16.1 Detailed Description

buffer fq output, write summary file

Author

Paula Perez paulaperezrubio@gmail.com

Date

06.10.2017

4.16.2 Function Documentation

4.16.2.1 void [buffer_outputDS](#) (FILE * *fout*, const char * *str*, const int *len*, const int *fd_i*)

buffers the output before writing to disk

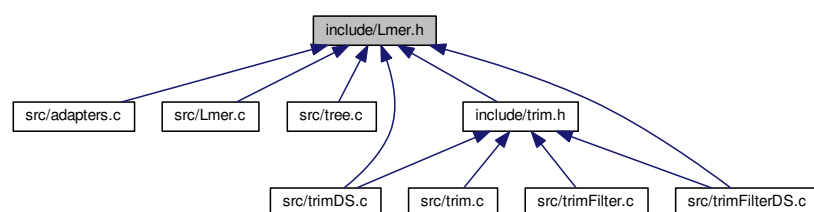
Parameters

| | |
|-------------|--|
| <i>fout</i> | FILE pointer where we might write to disk; |
| <i>str</i> | string we want to add |
| <i>len</i> | length of the string we want to add |
| <i>fd_i</i> | identifier: GOOD1, ADAP1, CONT1, LOWQ1, NNNN1, GOOD2, ADAP2, CONT2, LOWQ2, NNNN2 |

4.17 include/Lmer.h File Reference

Manipulation of Lmers and sequences.

This graph shows which files directly or indirectly include this file:



Functions

- void `init_map` ()
Initialize lookup table fw_1B.
- void `init_map_SA` ()
Initialize lookup table fw_1B (for SA)
- void `Lmer_sLmer` (char *Lmer, int L)
Transforms an Lmer to the convention stored in the lookup table fw_1B.
- void `rev_comp` (char *sLmer, int L)
Obtains the reverse complement, for {'\000','\001','\002','\003'}.
- void `rev_comp2` (char *sLmer, int L)
Obtains the reverse complement, for {'\001','\002','\003','\004'}.

4.17.1 Detailed Description

Manipulation of Lmers and sequences.

Author

Paula Perez paulaperezrubi@gmail.com

Date

18.08.2017

Note

I have to try to merge the two versions of conversions!

Basically, and depending on the method used, nucleotides {'a', 'c', 'g', 't'} are shifted to the characters {'\000','\001','\002','\003'} or to {'\001','\002','\003','\004'} in a Lmer. A function to provide the reverse complement is also provided.

4.17.2 Function Documentation

4.17.2.1 void `init_map` ()

Initialize lookup table fw_1B.

{'a','c','g','t'} → {'\000','\001','\002','\003'}, rest '\004'.

4.17.2.2 void `init_map_SA` ()

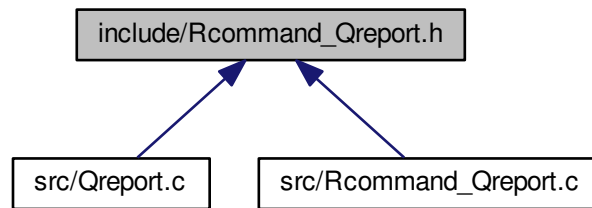
Initialize lookup table fw_1B (for SA)

{'a','c','g','t'} → {'\001','\002','\003','\004'}, rest '\005'.

4.18 include/Rcommand_Qreport.h File Reference

get Rscript command for Qreport

This graph shows which files directly or indirectly include this file:



Functions

- `char * command_Qreport ()`
returns Rscript command that generates the quality report in html

4.18.1 Detailed Description

get Rscript command for Qreport

Author

Paula Perez paulaperezrubio@gmail.com

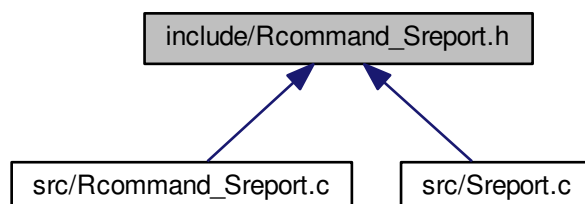
Date

09.08.2017

4.19 include/Rcommand_Sreport.h File Reference

get Rscript command for Sreport

This graph shows which files directly or indirectly include this file:



Functions

- char * [command_Sreport](#) ()

returns Rscript command that generates the summary report in html

4.19.1 Detailed Description

get Rscript command for Sreport

Author

Paula Perez paulaperezrubio@gmail.com

Date

09.08.2017

4.19.2 Function Documentation

4.19.2.1 char* [command_Sreport](#) ()

returns Rscript command that generates the summary report in html

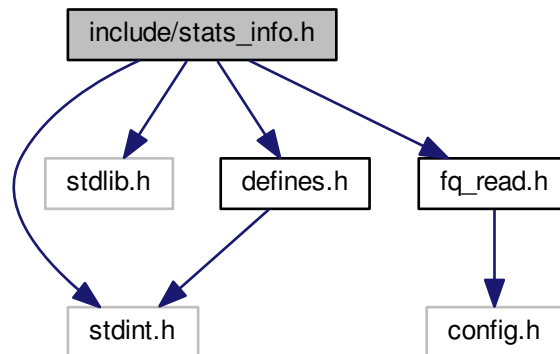
```
1 # To run between quotation marks after: Rscript_RBioC -e (Rscript)
2 inputfolder = normalizePath( <par.SR.inputfolder>, mustWork = TRUE);
3 output = <par_SR.outputfile>;
4 output_file = gsub('.* /', '', output);
5 path = gsub('[^/]+$' , '', output);
6 if (path != '') {
7   outputfile = paste0(normalizePath(path, mustWork = TRUE), '/', outputfile);
8 } else {
9   outputfile = paste0(cwd, '/', output_file); # cwd: current working dir
10 };
11 rmarkdown::render(<par_SR.Rmd_file>,
12                  params = list(inputfolder = inputfolder, version= VERSION),
13                  output_file = output_file)
```

4.20 include/stats_info.h File Reference

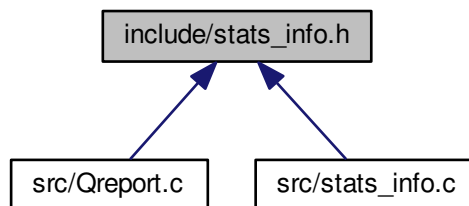
Construct the quality report variables and update them.

```
#include <stdint.h>
#include <stdlib.h>
#include "fq_read.h"
#include "defines.h"
```

Include dependency graph for stats_info.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [statsinfo](#)
stores info needed to create the summary graphs

Typedefs

- typedef struct [statsinfo](#) [Info](#)
stores info needed to create the summary graphs

Functions

- void [init_info](#) ([Info](#) *res)
Initialization of a Info type.
- void [free_info](#) ([Info](#) *res)
frees allocated memory in Info

- void `read_info` (`Info *res`, `char *file`)
Read Info from binary file.
- void `write_info` (`Info *res`, `char *file`)
Write info to binary file.
- void `print_info` (`Info *res`, `char *infofile`)
print Info to a textfile
- void `get_first_tile` (`Info *res`, `Fq_read *seq`)
gets first tile
- void `update_info` (`Info *res`, `Fq_read *seq`)
updates Info with Fq_read
- int `update_ACGT_counts` (`uint64_t *ACGT_low`, `char ACGT`)
update, for current tile, ACGT counts.
- void `update_QPosTile_table` (`Info *res`, `Fq_read *seq`)
update QPostile table
- void `update_ACGT_pos` (`uint64_t *ACGT_pos`, `Fq_read *seq`, `int read_len`)
update ACGT_pos
- void `resize_info` (`Info *res`)
resize Info

4.20.1 Detailed Description

Construct the quality report variables and update them.

Author

Paula Perez paulaperezrubio@gmail.com

Date

04.08.2017

4.20.2 Function Documentation

4.20.2.1 void init_info (Info * res)

Initialization of a Info type.

It sets: nQ, read_len, ntiles, minQ and the dimensions of the arrays. Initializes the rest of the variables to zero and allocates memory to the arrays initializing them to 0 (calloc).

4.20.2.2 void resize_info (Info * res)

resize Info

At the end of the program, resize the structure Info, and adapt it to the actual number of tiles and the actual number of different quality values present.

4.20.2.3 int update_ACGT_counts (uint64_t * ACGT_low, char ACGT)

update, for current tile, ACGT counts.

Makes update of ACGT counts for the current tile. Can be used with variables: lowQ_ACGT_tile and ACGT_tile

4.21 include/str_manip.h File Reference

functions that do string manipulation

This graph shows which files directly or indirectly include this file:



Classes

- [struct `_split`](#)
contains a splitted string and the number of splitted fields

Typedefs

- [typedef struct `_split` Split](#)
contains a splitted string and the number of splitted fields

Functions

- [int `str_isascii`](#) (char *s)
return nonzero iff all elements in the string are in the ASCII set.
- [int `strindex`](#) (char *s, char *t)
returns index of t in s (start, first occurrence)
- [int `count_char`](#) (char *s, char c)
returns the # of occurrences of char c in string s
- [Split `strsplit`](#) (char *str, char sep)
Separates strings by a separator.

4.21.1 Detailed Description

functions that do string manipulation

Author

Paula Perez paulaperezrubio@gmail.com

Date

03.08.2017

4.21.2 Function Documentation

4.21.2.1 int `strindex` (char * s, char * t)

returns index of t in s (start, first occurrence)

Parameters

| | |
|----------|-----------------------------|
| <i>s</i> | string to be checked. |
| <i>t</i> | substring to be found in s. |

4.21.2.2 Split strsplit (char * *str*, char *sep*)

Separates strings by a separator.

Parameters

| | |
|------------|------------------|
| <i>str</i> | input string |
| <i>sep</i> | separator (char) |

Returns

array of strings containing the substrings in the input separated

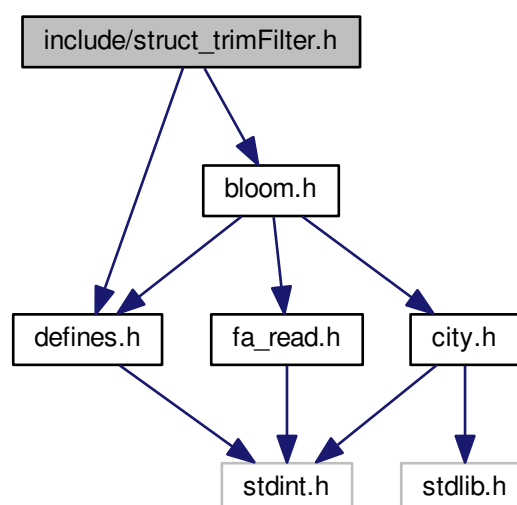
4.22 include/struct_trimFilter.h File Reference

structure where the input arguments of trimFilter and trimFilterDS will be stored and function to free the memory of it.

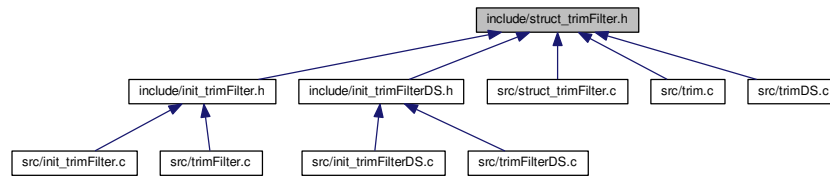
```
#include "defines.h"
```

```
#include "bloom.h"
```

Include dependency graph for struct_trimFilter.h:



This graph shows which files directly or indirectly include this file:



Classes

- [struct _adapter](#)
- [struct _iparam_trimFilter](#)
trimFilter input parameters

Typedefs

- typedef struct [_adapter](#) Adapter
- typedef struct [_iparam_trimFilter](#) Iparam_trimFilter
trimFilter input parameters

Functions

- void [free_parTF](#) ([Iparam_trimFilter](#) *ptr_parTF)
frees the allocated memory in Iparam_trimFilter

4.22.1 Detailed Description

structure where the input arguments of trimFilter and trimFilterDS will be stored and function to free the memory of it.

Author

Paula Perez paulaperezrubio@gmail.com

Date

07.10.2017

4.22.2 Typedef Documentation

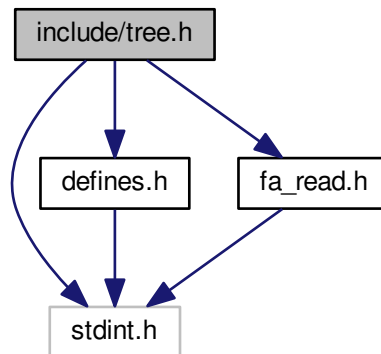
4.22.2.1 typedef struct _adapter Adapter

@ brief adapter struct @ note UNFINISHED!

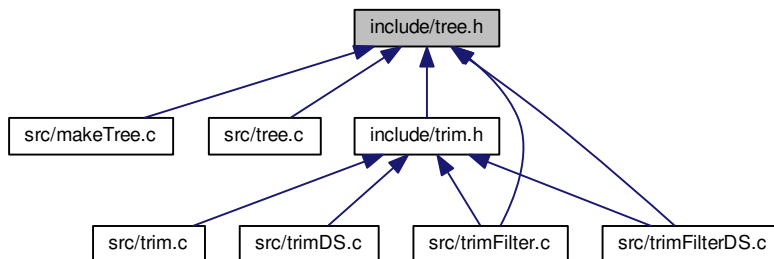
4.23 include/tree.h File Reference

Construction of tree, check paths, write tree, read in tree.

```
#include <stdint.h>
#include "defines.h"
#include "fa_read.h"
Include dependency graph for tree.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- `struct _node`
Node structure: formed out of `T_ACGT` pointers to Node structure.
- `struct _tree`
structure containing a `T_ACGT`-tree.

Typedefs

- `typedef struct _node Node`

Node structure: formed out of T_ACGT pointers to Node structure.

- `typedef struct _tree Tree`
structure containing a T_ACGT-tree.

Functions

- `Node * get_new_pool (Tree *tree_ptr)`
reallocs pool_2D (++NPOOL_2D) if all existing nodes have been used
- `Node * new_node_buf (Tree *tree_ptr)`
moves to the next node (allocating new memory if necessary)
- `void free_all_nodes (Tree *tree_ptr)`
frees the whole tree structure
- `void insert_Lmer (Tree *tree_ptr, char *Lmer)`
Lmer insertion in the tree (depth L).
- `void insert_entry (Tree *tree_ptr, Fa_entry *entry)`
fasta entry insertion in the tree (depth L).
- `double check_path (Tree *tree_ptr, char *read, int Lread)`
checks if read is found in tree and outputs a score
- `Tree * tree_from_fasta (Fa_data *fasta, int L)`
create Tree structure from fasta structure.
- `void save_tree (Tree *tree_ptr, char *filename)`
saves Tree to disk in filename
- `Tree * read_tree (char *filename)`
read tree from file

4.23.1 Detailed Description

Construction of tree, check paths, write tree, read in tree.

Author

Paula Perez paulaperezrubio@gmail.com

Date

18.08.2017

4.23.2 Typedef Documentation

4.23.2.1 `typedef struct _tree Tree`

structure containing a T_ACGT-tree.

The tree structure is stored in a pointer to pointer to Node. We grow the structure on the flight as we need more memory. In the outer direction, we start by allocating NPOOL_2D pointers to Node. In the inner direction, we allocate NPOOL_1D Nodes and fill them as we read the fasta file. When all of them are allocated, we allocate again NPOOL_1D. If NPOOL_2D pointers to Node are allocated, the outer dimension is reallocated with +NPOOL_2D extra elements. L is the depth of the tree, pool_count is the number on Node* elements used so far, pool_available is the number of Nodes available in every moment, and nnodes is the total number of nodes filled in. We limit the number of allocated nodes to UINT_MAX (we cannot count more nodes!).

4.23.3 Function Documentation

4.23.3.1 double check_path (Tree * *tree_ptr*, char * *read*, int *Lread*)

checks if read is found in tree and outputs a score

Parameters

| | |
|-----------------|----------------------------|
| <i>tree_ptr</i> | pointer to Tree structure |
| <i>read</i> | Read or reverse complement |
| <i>Lread</i> | length of read |

Returns

score = (number of Lmers of reads found in read) / (Lread-L+1)

4.23.3.2 void free_all_nodes (Tree * tree_ptr)

frees the whole tree structure

Parameters

| | |
|-----------------|---------------------------|
| <i>tree_ptr</i> | pointer to Tree structure |
|-----------------|---------------------------|

This function deallocates the memory allocated in a Tree structure.

4.23.3.3 Node* get_new_pool (Tree * tree_ptr)

reallocs pool_2D (++NPOOL_2D) if all existing nodes have been used

Parameters

| | |
|-----------------|---------------------------|
| <i>tree_ptr</i> | pointer to Tree structure |
|-----------------|---------------------------|

4.23.3.4 Node* new_node_buf (Tree * tree_ptr)

moves to the next node (allocating new memory if necessary)

Parameters

| | |
|-----------------|---------------------------|
| <i>tree_ptr</i> | pointer to Tree structure |
|-----------------|---------------------------|

Returns

address to next node

The function checks if there are available nodes (information stored in the variable *tree_ptr* -> *pool_available*) and goes to the next node. If there is no nodes left, it allocates a new pool_1D, and if there is no room left in the outer dimension, it reallocates NPOOL_2D more Node*'s. If the number of nodes reaches UINT_MAX, the program returns an error message and exits.

4.23.3.5 Tree* read_tree (char * filename)

read tree from file

Parameters

| | |
|-----------------|--------------------------|
| <i>filename</i> | string with the filename |
|-----------------|--------------------------|

Returns

pointer to Tree structure

This function unwinds the process carried out in *save_tree* and assigns addresses to the children of every given node.

4.23.3.6 void save_tree (Tree * *tree_ptr*, char * *filename*)

saves Tree to disk in filename

Parameters

| | |
|-----------------|----------------------------|
| <i>tree_ptr</i> | pointer to Tree structure |
| <i>filename</i> | string containing filename |

The tree structure is stored as follows: every address is stored in a uint32_t (we are not allowing trees with more than UINT_MAX nodes). For every node, the addresses of the children are stored in the following fashion:

- If it is pointing to NULL: 0.
- Otherwise: i2, the index in the outer dimension of pool_2D is identified, and the difference $\text{jump} = \text{pool_2D}[i][j].\text{children}[k] - \text{pool_2D}[i2]$ is computed. $i2 * \text{NPOOL_D1} + \text{jump}$ is then stored for child k.

4.23.3.7 Tree* tree_from_fasta (Fa_data * fasta, int L)

create Tree structure from fasta structure.

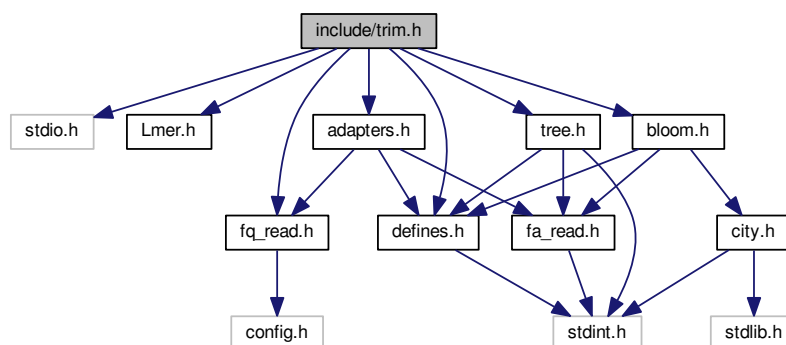
Parameters

| | |
|--------------|----------------------------|
| <i>fasta</i> | pointer to fasta structure |
| <i>L</i> | tree length |

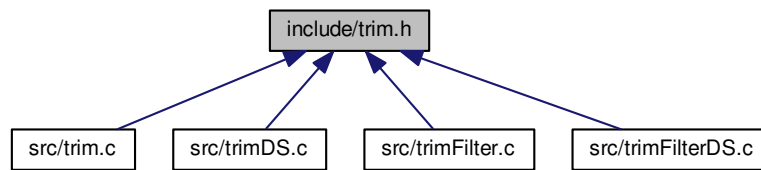
4.24 include/trim.h File Reference

trims/filter sequences after Quality, N's contaminations.

```
#include <stdio.h>
#include "Lmer.h"
#include "fq_read.h"
#include "defines.h"
#include "tree.h"
#include "bloom.h"
#include "adapters.h"
Include dependency graph for trim.h:
```



This graph shows which files directly or indirectly include this file:



Functions

- `int trim_adapter (Fq_read *seq, Ad_seq *adap_list)`
trims sequence based on presence of N nucleotides
- `int trim_sequenceN (Fq_read *seq)`
trims sequence based on presence of N nucleotides
- `int trim_sequenceQ (Fq_read *seq)`
trims sequence based on lowQ base callings
- `bool is_read_inTree (Tree *tree_ptr, Fq_read *seq)`
check if Lread is contained in tree. It computes the score for the read and its reverse complement; if one of them exceeds the user selected threshold, it returns true. Otherwise, it returns false.
- `bool is_read_inBloom (Bfilter *tree_ptr, Fq_read *seq, Bfkmer *ptr_Bfkmer)`
checks if a read is in Bloom filter. It computes the score for the read and returns true if it exceeds the user selected threshold. Returns false otherwise.
- `int Qtrim_global (Fq_read *seq, int left, int right, char type)`
trims left from the left and right from the right

4.24.1 Detailed Description

trims/filter sequences after Quality, N's contaminations.

Author

Paula Perez paulaperezrubio@gmail.com

Date

24.08.2017

4.24.2 Function Documentation

4.24.2.1 `bool is_read_inBloom (Bfilter * ptr_bf, Fq_read * seq, Bfkmer * ptr_bfkmer)`

checks if a read is in Bloom filter. It computes the score for the read and returns true if it exceeds the user selected threshold. Returns false otherwise.

Parameters

| | |
|------------------|---|
| <i>ptr_bf</i> | pointer to Bfilter |
| <i>seq</i> | fastq read |
| <i>ptr_bfkmr</i> | pointer to Procs_kmer structure (will store global) |

Returns

true if read was found, false otherwise

4.24.2.2 **bool** is_read_inTree (**Tree** * *tree_ptr*, **Fq_read** * *seq*)

check if Lread is contained in tree. It computes the score for the read and its reverse complement; if one of them exceeds the user selected threshold, it returns true. Otherwise, it returns false.

Parameters

| | |
|-----------------|---------------------------|
| <i>tree_ptr</i> | pointer to Tree structure |
| <i>seq</i> | fastq read |

Returns

true if read was found, false otherwise

4.24.2.3 **int** Qtrim_global (**Fq_read** * *seq*, **int** *left*, **int** *right*, **char** *type*)

trims left from the left and right from the right

Parameters

| | |
|--------------|--|
| <i>seq</i> | fastq read |
| <i>left</i> | number of nucleotides to be trimmed from the left |
| <i>right</i> | number of nucleotides to be trimmed from the right |
| <i>type</i> | char indicating the type of trimming (Q,A). |

Returns

2, since they are all accepted and trim

4.24.2.4 **int** trim_adapter (**Fq_read** * *seq*, **Ad_seq** * *adap_list*)

trims sequence based on presence of N nucleotides

if (adapter length < 16) -> search for seeds 8 nucleotides long else -> search for seeds 16 nucleotides long if (seed found) -> calculate score if score > threshold -> aligner found, trim / discard and exit. else -> search for seeds 8 nucleotides long

Parameters

| | |
|------------------|---------------------------|
| <i>seq</i> | pointer to Fq_read |
| <i>adap_list</i> | array of Ad_seq |

Returns

-1 error, 0 discarded, 1 accepted as is, 2 accepted and trimmed

Note

Global input parameters from par_TF are also used

4.24.2.5 int trim_sequenceN (Fq_read * seq)

trims sequence based on presence of N nucleotides

Parameters

| | |
|-----|------------|
| seq | fastq read |
|-----|------------|

Returns

-1 error, 0 discarded, 1 accepted as is, 2 accepted and trimmed

This function calls a different function depending on the method passed as input par_TF.trimN:

- [NO\(0\)](#): accepts it as is, (1),
- [ALL\(1\)](#): accepts it as is if NO N's found (1), rejects it otherwise (0),
- [ENDS\(2\)](#): trims the ends and accepts it if it is longer than minL (2 if trimming, 1 if no trimming), rejects it otherwise (0),
- [STRIP\(3\)](#): finds the longest N-free subsequence and trims it if it is at least minL nucleotides long (2 if trimming, 1 if no N's are found), rejects it otherwise (0).

4.24.2.6 int trim_sequenceQ (Fq_read * seq)

trims sequence based on lowQ base callings

Parameters

| | |
|-----|------------|
| seq | fastq read |
|-----|------------|

Returns

-1 error, 0 discarded, 1 accepted as is, 2 accepted and trimmed

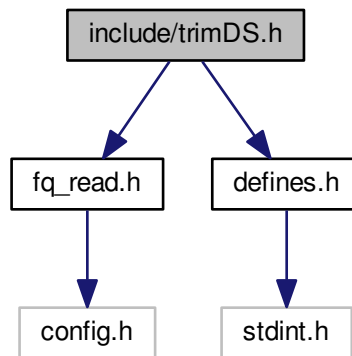
This function calls a different function depending on the method passed as input par_TF.trimQ:

- [NO\(0\)](#): accepts is as is , (1),
- [FRAC\(1\)](#): accepts it if less than par_TF.nlowQ are found (1), rejects it otherwise (0),
- [ENDS\(2\)](#): trims the ends and accepts it if it is longer than minL (2 if trimming, 1 if no trimming), rejects it otherwise (0),
- [ENDSFRAC\(3\)](#): trims the ends and accepts if the remaining sequence is at least minL bases long and if it contains less than nlowQ lowQ nucleotides (2 if trimming, 1 if no trimming). Otherwise, it is rejected, (0).
- [GLOBAL\(4\)](#): it trims globally globleft nucleotides from the left and globright from the right, (returns 2).

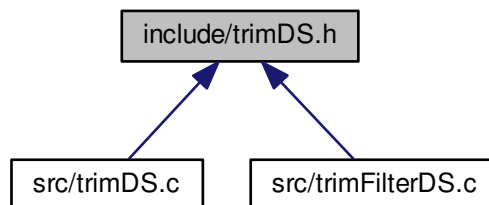
4.25 include/trimDS.h File Reference

trim adapters from double stranded data

```
#include "fq_read.h"
#include "defines.h"
Include dependency graph for trimDS.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- `struct _ds_adap`
structure containing an adapter pair (for read 1 and read 2)

Typedefs

- `typedef struct _ds_adap DS_adap`
structure containing an adapter pair (for read 1 and read 2)

Functions

- `DS_adap init_DSadap (char *ad1, char *ad2, int L1, int L2)`
initialization of a DS_adap structure

- int trim_adapterDS (DS_adap *ptr_DSad, Fq_read *r1, Fq_read *r2)

trim the sequences, discard them or keep them unchanged depending on them having adapters remnants.

4.25.1 Detailed Description

trim adapters from double stranded data

Author

Paula Perez paulaperezrubio@gmail.com

Date

05.10.2017

4.25.2 Function Documentation

4.25.2.1 DS_adap init_DSadap (char * ad1, char * ad2, int L1, int L2)

initialization of a DS_adap structure

Parameters

| | |
|------------|---------------------------|
| <i>ad1</i> | adapter 1 sequence |
| <i>ad2</i> | adapter 2 sequence |
| <i>L1</i> | adapter 1 sequence length |
| <i>L2</i> | adapter 2 sequence length |

Returns

initialized DS_adap structure

4.25.2.2 int trim_adapterDS (DS_adap * ptr_DSad, Fq_read * r1, Fq_read * r2)

trim the sequences, discard them or keep them unchanged depending on them having adapters remnants.

Returns

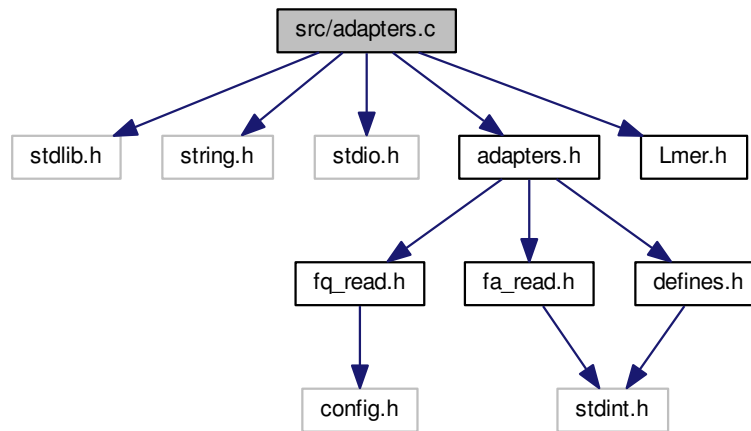
0 if the read is to be discarded, 1 if left as is, 2 if trimmed. Reads are trimmed if 2 is returned.

4.26 src/adapters.c File Reference

sequence manipulation for alignment

```
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include "adapters.h"
#include "Lmer.h"
```

Include dependency graph for adapters.c:



Functions

- void `init_alUTs` ()
look up table initialization for alignment (used for adapters)
- int `process_seq` (unsigned char *packed, unsigned char *sequence, int L, bool shift, bool isreverse)
Packs a sequence using alfw0, alfw1, albw0, albw1.
- `Ad_seq` * `pack_adapter` (`Fa_data` *ptr_fa)
*reads a **Fa_data** with adapters and stores them in an array of **Ad_seq** structs.*
- double `obtain_score` (`Fq_read` *seq, int pos_seq, `Ad_seq` *ptr_adap, int pos_ad)
computes score of a possible alignment, after having found a seed.

Variables

- static uint8_t `alfw0` [256]
- static uint8_t `alfw1` [256]
- static uint8_t `albw0` [256]
- static uint8_t `albw1` [256]
- uint8_t `fw_1B` [256]
- uint8_t `bw_1B` [256]

4.26.1 Detailed Description

sequence manipulation for alignment

Author

Paula Perez paulaperezrubio@gmail.com

Date

23.09.2017

4.26.2 Function Documentation

4.26.2.1 void init_alLUTs ()

look up table initialization for alignment (used for adapters)

It initializes: fw_1B, bw_1B. They are uint8_t arrays with 256 elements. All elements are set to 0xFF excepting the ones corresponding to 'a', 'A', 'c', 'C', 'g', 'G', 't', 'T':

| Var | a,A | c,C | g,G | t,T | Var | a,A | c,C | g,G | t,T |
|-------|------|------|------|------|-------|------|------|------|------|
| alfw0 | 0x01 | 0x02 | 0x04 | 0x08 | albw0 | 0x08 | 0x04 | 0x02 | 0x01 |
| alfw1 | 0x10 | 0x20 | 0x40 | 0x80 | albw1 | 0x80 | 0x40 | 0x20 | 0x10 |

With this variables we will encode sequences that can be compared later on. Using the bitwise XOR operator, every mismatch will amount to two bits set to 1.

4.26.2.2 double obtain_score (Fq_read * seq, int pos_seq, Ad_seq * ptr_adap, int pos_ad)

computes score of a possible alignment, after having found a seed.

The score is computed as follows:

- matching bases: score += log₁₀(4)
- unmatching bases: score -= Q/10, where Q is the quality score.

Parameters

| | |
|-----------------|--|
| <i>seq</i> | pointer to Fq_read . |
| <i>pos_seq</i> | read starting position of the alignment |
| <i>ptr_adap</i> | pointer to Ad_seq , contains the adapter info |
| <i>pos_ad</i> | adapter starting position of the alignment (reverse) |

Returns

score of the alignment

4.26.2.3 Ad_seq* pack_adapter (Fa_data * ptr_fa)

reads a **Fa_data** with adapters and stores them in an array of **Ad_seq** structs.

It reads the fasta structure. For every entry, an **Ad_seq** structure is allocated and the sequences are processed to create the packed sequences.

Parameters

| | |
|---------------|-------------------------------------|
| <i>ptr_fa</i> | pointer to Fa_data structure |
|---------------|-------------------------------------|

Returns

pointer to **Ad_seq**, where the information is stored.

4.26.2.4 int process_seq (unsigned char * packed, unsigned char * sequence, int L, bool shift, bool isreverse)

Packs a sequence using alfw0, alfw1, albw0, albw1.

It takes a sequence of length L and packs it using the look up tables into an unsigned char array, where every bytes corresponds to 2 nucleotides. One can encode the reverse complement or the sequence shifted by 1/2 byte.

Parameters

| | |
|------------------|---|
| <i>packed</i> | packed sequence |
| <i>sequence</i> | original sequence |
| <i>L</i> | original sequence length |
| <i>shift</i> | 0 if taken as is we want to shift the output 1/2 byte (>>4) |
| <i>isreverse</i> | 0 if we want the forward sequence, 1 reverse complement |

Returns

Lhalf, length in Bytes of the packed sequence

4.26.3 Variable Documentation**4.26.3.1** `uint8_t albw0[256]` `[static]`

variable for brackward packing, first half

4.26.3.2 `uint8_t albw1[256]` `[static]`

variable for brackward packing, second half

4.26.3.3 `uint8_t alfw0[256]` `[static]`

variable for forward packing, first half

4.26.3.4 `uint8_t alfw1[256]` `[static]`

variable for forward packing, second half

4.26.3.5 `uint8_t bw_1B[256]`

global variable. Lookup table.

4.26.3.6 `uint8_t fw_1B[256]`

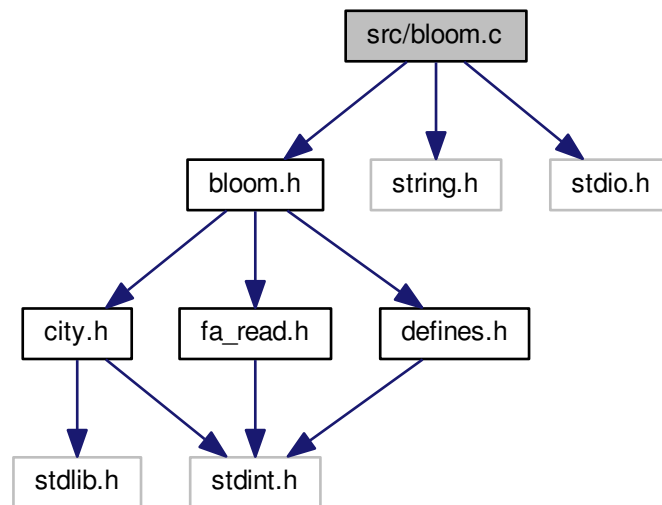
global variable. Lookup table.

4.27 src/bloom.c File Reference

functions that implement the bloom filter

```
#include "bloom.h"
#include <string.h>
#include <stdio.h>
```

Include dependency graph for bloom.c:



Functions

- void `init_LUTs` ()
look up table initialization
- `Bfilter` * `init_Bfilter` (int kmersize, uint64_t bsizeBits, int hashNum, double falsePosRate, uint64_t nelemt)
initialization of a Bfilter structure
- void `free_Bfilter` (`Bfilter` *ptr_bf)
free Bfilter memory
- `Bfkmer` * `init_Bfkmer` (int kmersize, int hashNum)
initializes a Bfkmer structure, given the kmersize and the number of hash functions
- void `free_Bfkmer` (`Bfkmer` *ptr_bfkmer)
free Bfkmer
- int `compact_kmer` (const unsigned char *sequence, uint64_t position, `Bfkmer` *ptr_bfkmer)
compactifies a kmer for insertion in the bloomfilter
- void `multiHash` (`Bfkmer` *ptr_bfkmer)
obtains the hashNum hashvalues for a compactified kmer
- bool `insert_and_fetch` (`Bfilter` *ptr_bf, `Bfkmer` *ptr_bfkmer)
inserts the hashvalues of a kmer in filter
- bool `contains` (`Bfilter` *ptr_bf, `Bfkmer` *ptr_bfkmer)
check if kmer is contained in the filter
- `Bfilter` * `create_Bfilter` (`Fa_data` *ptr_fasta, int kmersize, uint64_t bsizeBits, int hashNum, double falsePosRate, uint64_t nelemt)
creates a bloom filter from a fasta structure.
- void `save_Bfilter` (`Bfilter` *ptr_bf, char *filterfile, char *paramfile)
saves a bloomfilter to disk
- `Bfilter` * `read_Bfilter` (char *filterfile, char *paramfile)
reads a bloom filter from a file

Variables

- static uint8_t **fw0** [256]
Global variables (lookup table) Used to compactify kmers.
- static uint8_t **fw1** [256]
- static uint8_t **fw2** [256]
- static uint8_t **fw3** [256]
- static uint8_t **bw0** [256]
- static uint8_t **bw1** [256]
- static uint8_t **bw2** [256]
- static uint8_t **bw3** [256]
- uint64_t **alloc_mem**

4.27.1 Detailed Description

functions that implement the bloom filter

Author

Paula Perez paulaperezrubio@gmail.com

Date

04.09.2017

4.27.2 Function Documentation

4.27.2.1 int compact_kmer (const unsigned char * *sequence*, uint64_t *position*, Bfkmer * *ptr_bfkmer*)

compactifies a kmer for insertion in the bloomfilter

Parameters

| | |
|-------------------|--|
| <i>sequence</i> | unsigned char DNA sequence (or cDNA) |
| <i>position</i> | position in the sequence where the kmer starts |
| <i>ptr_bfkmer</i> | initialized Bfkmer |

The compactified sequence is computed in the following way:

- We start compactifying both, the forward and backward (reverse complement). The outer loop covers up until half of the sequence.
- As soon as one of the two is lexicographically smaller, we continue only with it. In that way, the "smaller" sequence is consistently returned.
- If the sequence is palindromic, we continue with the forward sequence.
- kmersize should be > 3.

We illustrate the compactification with an example:

```
kmer = TTTT|GGAT
m_fw = 00000000 | 00000000 // 2 bytes
m_bw = 00000000 | 00000000 // 2 bytes
m_fw[0] |= fw0['T'] = 0xC0|0x00; m_bw[0] |= bw0['T'] = 0x00|0x00;
m_fw[0] |= fw1['T'] = 0xF0|0x00; m_bw[0] |= bw1['A'] = 0x30|0x00;
m_fw[0] |= fw2['T'] = 0xFC|0x00; m_bw[0] |= bw2['G'] = 0x34|0x00;
m_fw[0] |= fw3['T'] = 0xFF|0x00; m_bw[0] |= bw3['G'] = 0x35|0x00;
m_fw[1] |= fw0['G'] = 0xC0|0x80; m_bw[1] |= bw0['T'] = 0x35|0x00;
m_fw[1] |= fw1['G'] = 0xF0|0xA0; m_bw[1] |= bw1['T'] = 0x35|0x00;
m_fw[1] |= fw2['A'] = 0xFC|0xA0; m_bw[1] |= bw2['T'] = 0x35|0x00;
m_fw[1] |= fw3['T'] = 0xFF|0xA3; m_bw[1] |= bw3['T'] = 0x35|0x00;
```

(In this case, we would store m_bw)

4.27.2.2 bool contains (Bfilter * ptr_bf, Bfkmr * ptr_bfkmr)

check if kmer is contained in the filter

Parameters

| | |
|-------------------|---|
| <i>ptr_bf</i> | pointer to a Bfilter structure, where a bloomfilter is stored |
| <i>ptr_bfkmer</i> | pointer to a Bfkmer structure containing the hash values |

Returns

true if all corresponding bits were set to 1 in the filter

4.27.2.3 Bfilter* create_Bfilter (Fa_data * ptr_fasta, int kmerSize, uint64_t bsizeBits, int hashNum, double falsePosRate, uint64_t nelem)

creates a bloom filter from a fasta structure.

Parameters

| | |
|---------------------|--|
| <i>ptr_fasta</i> | pointer to fasta structure |
| <i>kmerSize</i> | length of kmers to be inserted in the filter |
| <i>bsizeBits</i> | size of Bloom filter in bits |
| <i>hashNum</i> | number of hash functions to be used |
| <i>falsePosRate</i> | false positive rate |
| <i>nelem</i> | number of elemens (kmers in the sequece) contained in the filter |

Returns

pointer to Bloom filter structure, where the fasta file was encoded.

4.27.2.4 Bfilter* init_Bfilter (int kmerSize, uint64_t bsizeBits, int hashNum, double falsePosRate, uint64_t nelem)

initialization of a Bfilter structure

Parameters

| | |
|---------------------|--|
| <i>kmerSize</i> | number of elements of the kmer |
| <i>bsizeBits</i> | size of the bloomfilter (in Bits) |
| <i>hashNum</i> | number of hash functions to be computed |
| <i>falsePosRate</i> | false positive rate |
| <i>nelem</i> | number of elemens (kmers in the sequece) contained in the filter |

Returns

pointer to initialized Bfilter structure

Given a kmerSize, bsizeBits, number of hash functions, we assign these values to the struture and the two additional values: kmerSizeBytes = (kmerSize + BASESINCHAR - 1)/BASESINCHAR

4.27.2.5 Bfkmer* init_Bfkmer (int kmerSize, int hashNum)

initializes a Bfkmer structure, given the kmerSize and the number of hash functions

Parameters

| | |
|-----------------|--------------------------------|
| <i>kmerSize</i> | number of elements of the kmer |
|-----------------|--------------------------------|

| | |
|----------------|---|
| <i>hashNum</i> | number of hash functions to be computed |
|----------------|---|

Returns

pointer to a Bfkmer structure

kmersizeBytes, halfsizeBytes, hangingBases, hasOverhead hashNum are assigned and memory is allocated and set to 0 for compact and hashValues

4.27.2.6 void init_LUTs ()

look up table initialization

It initializes: fw0, fw1, fw2, fw3, bw0, bw2, bw3, bw4. They are uint8_t arrays with 256 elements. All elements are set to 0xFF excepting the ones corresponding to 'a', 'A', 'c', 'C', 'g', 'G', 't', 'T':

| Var | a,A | c,C | g,G | t,T | Var | a,A | c,C | g,G | t,T |
|-----|------|------|------|------|-----|------|------|------|------|
| fw0 | 0x00 | 0x40 | 0x80 | 0xC0 | bw0 | 0xC0 | 0x80 | 0x40 | 0x00 |
| fw1 | 0x00 | 0x10 | 0x20 | 0x30 | bw1 | 0x30 | 0x20 | 0x10 | 0x00 |
| fw2 | 0x00 | 0x04 | 0x08 | 0x0C | bw2 | 0x0C | 0x08 | 0x04 | 0x00 |
| fw3 | 0x00 | 0x01 | 0x02 | 0x03 | bw3 | 0x03 | 0x02 | 0x01 | 0x00 |

With these variables, we will be able to encode a Sequence using 2 bits per nucleotide.

4.27.2.7 bool insert_and_fetch (Bfilter * ptr_bf, Bfkmer * ptr_bfkmer)

inserts the hashvalues of a kmer in filter

Parameters

| | |
|-------------------|---|
| <i>ptr_bf</i> | pointer to Bfilter structure, where we will include the new entry |
| <i>ptr_bfkmer</i> | pointer to Bfkmer structure, where the hashvalues are stored |

Returns

true if the positions of the hash values were already set to one previously.

The hash values are inserted in the following way.

- modValue = hashvalue mod(filter size) is calculated.
- the bit in position modValue of the filter is set to 1.

4.27.2.8 void multiHash (Bfkmer * ptr_bfkmer)

obtains the hashNum hashvalues for a compactified kmer

The hash values are computed using the CityHash64 hash functions.

4.27.2.9 Bfilter* read_Bfilter (char * filterfile, char * paramfile)

reads a bloom filter from a file

Parameters

| | |
|-------------------|------------------------------------|
| <i>filterfile</i> | path to file containing the filter |
| <i>paramfile</i> | path to file containing the filter |

Returns

a pointer to a filter structure containing the bloomfilter

This function reads two files, the auxiliar inputfile where kmersize, hashNum and bsizeBits are stored, and the actual filter file. If one of them is missing, the program exits with an error. If successful, a pointer to a Bfilter structure with the bloom filter is return

4.27.2.10 void save_Bfilter (Bfilter * ptr_bf, char * filterfile, char * paramfile)

saves a bloomfilter to disk

Parameters

| | |
|-------------------|--|
| <i>ptr_bf</i> | pointer to Bfilter structure (contains the filter) |
| <i>filterfile</i> | path to file where the output will be stored |
| <i>paramfile</i> | path to file where the prameters will be stored |

This function will save the bloomfilter in the path filterfile. The paramfile will store the following data:

- kmersize
- hashNum
- bsizeBits
- falsePosRate
- nelem

4.27.3 Variable Documentation

4.27.3.1 uint64_t alloc_mem

allocated memory

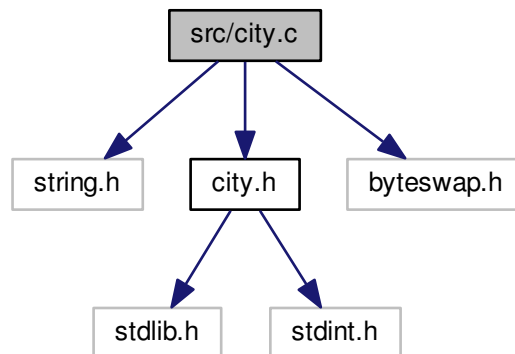
global variable. Memory allocated in the heap.

4.28 src/city.c File Reference

functions for hashin strings, C translation of cityhash (C++, google)

```
#include <string.h>
#include "city.h"
#include <byteswap.h>
```

Include dependency graph for city.c:



Macros

- `#define uint32_in_expected_order(x) (x)`
- `#define uint64_in_expected_order(x) (x)`
- `#define LIKELY(x) (x)`
- `#define PERMUTE3(a, b, c) do { std::swap(a, b); std::swap(a, c); } while (0)`
- `#define PERMUTE3_32(a, b, c) do { uint32_t t = a; a = c; c = b; b = t; } while (0)`
- `#define PERMUTE3_64(a, b, c) do { uint64_t t = a; a = c; c = b; b = t; } while (0)`

Functions

- static uint64 **UNALIGNED_LOAD64** (const char *p)
- static uint32 **UNALIGNED_LOAD32** (const char *p)
- static uint64 **Fetch64** (const char *p)
- static uint32 **Fetch32** (const char *p)
- static uint32 **fmix** (uint32 h)
- static uint32 **Rotate32** (uint32 val, int shift)
- static uint32 **Mur** (uint32 a, uint32 h)
- static uint32 **Hash32Len13to24** (const char *s, size_t len)
- static uint32 **Hash32Len0to4** (const char *s, size_t len)
- static uint32 **Hash32Len5to12** (const char *s, size_t len)
- uint32 **CityHash32** (const char *s, size_t len)
- static uint64 **Rotate** (uint64 val, int shift)
- static uint64 **ShiftMix** (uint64 val)
- static uint64 **HashLen16** (uint64 u, uint64 v)
- static uint64 **HashLen16_3a** (uint64 u, uint64 v, uint64 mul)
- static uint64 **HashLen0to16** (const char *s, size_t len)
- static uint64 **HashLen17to32** (const char *s, size_t len)
- [uint128 WeakHashLen32WithSeeds](#) (uint64 w, uint64 x, uint64 y, uint64 z, uint64 a, uint64 b)
- [uint128 WeakHashLen32WithSeeds_3a](#) (const char *s, uint64 a, uint64 b)
- static uint64 **HashLen33to64** (const char *s, size_t len)
- uint64 **CityHash64** (const char *s, size_t len)
- uint64 **CityHash64WithSeed** (const char *s, size_t len, uint64 seed)

- uint64 **CityHash64WithSeeds** (const char *s, size_t len, uint64 seed0, uint64 seed1)
- static uint128 **CityMurmur** (const char *s, size_t len, uint128 seed)
- uint128 **CityHash128WithSeed** (const char *s, size_t len, uint128 seed)
- uint128 **CityHash128** (const char *s, size_t len)

Variables

- static const uint64 **k0** = 0xc3a5c85c97cb3127ULL
- static const uint64 **k1** = 0xb492b66fbe98f273ULL
- static const uint64 **k2** = 0x9ae16a3b2f90404fULL
- static const uint32_t **c1** = 0xcc9e2d51
- static const uint32_t **c2** = 0x1b873593

4.28.1 Detailed Description

functions for hashin strings, C translation of cityhash (C++, google)

Author

bdnt

See also

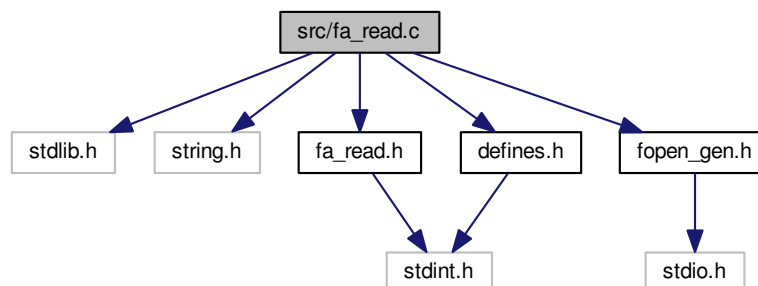
<https://github.com/bdnt/cityhash-c>
<https://github.com/google/cityhash>

4.29 src/fa_read.c File Reference

reads in and stores fasta files

```
#include <stdlib.h>
#include <string.h>
#include "fa_read.h"
#include "defines.h"
#include "fopen_gen.h"
```

Include dependency graph for fa_read.c:



Functions

- static int `ignore_line` (char *line)
ignore header lines.
- static void `init_fa` (Fa_data *ptr_fa)
Initialization of Fa_data.
- static void `realloc_fa` (Fa_data *ptr_fa)
Reallocation of Fa_data, in case the length of entrylen is exhausted.
- static void `init_entries` (Fa_data *ptr_fa)
Allocation of Fa_entries.
- static uint64_t `sweep_fa` (char *filename, Fa_data *ptr_fa)
this function sweeps a fasta file to obtain structure details.
- int `read_fasta` (char *filename, Fa_data *ptr_fa)
reads a fasta file and stores the contents in a Fa_data structure.
- uint64_t `size_fasta` (Fa_data *ptr_fa)
computes length of genome in fasta structure
- uint64_t `nkmers` (Fa_data *ptr_fa, int kmersize)
number of kmers of length kmersize contained in a fasta structure
- void `free_fasta` (Fa_data *ptr_fa)
free fasta file

Variables

- uint64_t `alloc_mem`

4.29.1 Detailed Description

reads in and stores fasta files

Author

Paula Perez paulaperezrubio@gmail.com

Date

18.08.2017

4.29.2 Function Documentation

4.29.2.1 void free_fasta (Fa_data * ptr_fa)

free fasta file

Parameters

| | |
|---------------------|-------------------------------|
| <code>ptr_fa</code> | pointer to Fa_data structure. |
|---------------------|-------------------------------|

The dynamically allocated memory in a Fa_data struct is deallocated and counted, so that we can

4.29.2.2 static int ignore_line (char * line) [static]

ignore header lines.

Parameters

| | |
|-------------|-----------------------|
| <i>line</i> | string of characters. |
|-------------|-----------------------|

Returns

number of characters to jump until a
is found.

4.29.2.3 static void init_entries (Fa_data * ptr_fa) [static]

Allocation of Fa_entries.

Parameters

| | |
|---------------|-------------------------------|
| <i>ptr_fa</i> | pointer to Fa_data structure. |
|---------------|-------------------------------|

When we have swept the fasta file once, we can proceed to allocate the memory for the entries (now we have registered their length).

4.29.2.4 static void init_fa (Fa_data * ptr_fa) [static]

Initialization of Fa_data.

Parameters

| | |
|---------------|-------------------------------|
| <i>ptr_fa</i> | pointer to Fa_data structure. |
|---------------|-------------------------------|

Initializes nlines, linelen, nentries to 0 and allocates memory for entrylen (FA_ENTRY_BUF entries).

4.29.2.5 uint64_t nkmers (Fa_data * ptr_fa, int kmersize)

number of kmers of length kmersize contained in a fasta structure

Returns

number of kmers of length kmersize contained in a fasta structure

4.29.2.6 int read_fasta (char * filename, Fa_data * ptr_fa)

reads a fasta file and stores the contents in a Fa_data structure.

Parameters

| | |
|-----------------|-------------------------------|
| <i>filename</i> | path to a fasta input file. |
| <i>ptr_fa</i> | pointer to Fa_data structure. |

Returns

number of entries in the fasta file.

A fasta file is read and stored in a structure Fa_data. The basic problem with reading FASTA files is that there is no end-of-record indicator. When you're reading sequence n, you don't know you're done until you've read the header line for sequence n+1, which you won't parse 'til later (when you're reading in the sequence n+1). The solution implemented here is to read the file twice. The first time, (sweep_fa), we initialize Fa_data and store the parameters:

- nlines: number of lines of the fasta file.

- nentries: number of entries in the fasta file.
- linelen: length of a line in the considered fasta file.
- entrylen: array containing the lengths of every entry. With this information, the pointer to Fa_entry can be allocated and the file is read again and the entries are stored in the structure.

4.29.2.7 static void realloc_fa (Fa_data * ptr_fa) [static]

Reallocation of Fa_data, in case the length of entrylen is exhausted.

Parameters

| | |
|---------------|-------------------------------|
| <i>ptr_fa</i> | pointer to Fa_data structure. |
|---------------|-------------------------------|

4.29.2.8 uint64_t size_fasta (Fa_data * ptr_fa)

computes length of genome in fasta structure

Parameters

| | |
|---------------|--------------------|
| <i>ptr_fa</i> | pointer to Fa_data |
|---------------|--------------------|

Returns

total number of nucleotides

4.29.2.9 static uint64_t sweep_fa (char * filename, Fa_data * ptr_fa) [static]

this function sweeps a fasta file to obtain structure details.

Parameters

| | |
|-----------------|-------------------------------|
| <i>filename</i> | path to a fasta input file. |
| <i>ptr_fa</i> | pointer to Fa_data structure. |

Returns

size of fasta file.

This function sweeps over the fasta file once to annotate how many entries there are, how long they are, how many characters there are per line, and how many lines the file has.

4.29.3 Variable Documentation

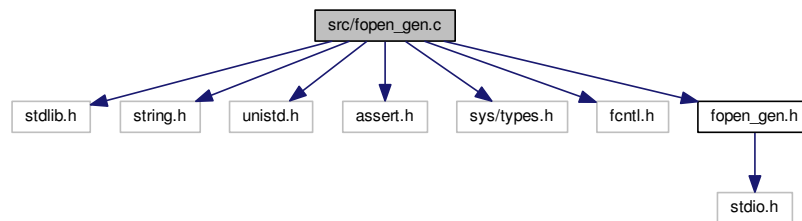
4.29.3.1 uint64_t alloc_mem

global variable. Memory allocated in the heap.

4.30 src/fopen_gen.c File Reference

Uncompress/compress input/output files using pipes.

```
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <assert.h>
#include <sys/types.h>
#include <fcntl.h>
#include "fopen_gen.h"
Include dependency graph for fopen_gen.c:
```



Functions

- static const char * **zcatExec** (const char *path)
- static const char * **catExec** (const char *path)
Commands to compress files. To be done in output.
- static int **uncompress** (const char *path)
Open a pipe to uncompress file. Open a pipe to uncompress the specified file. Not thread safe.
- static int **compress** (const char *path)
Open a pipe to compress output. Open a pipe to uncompress the specified file. Not thread safe.
- int **setCloexec** (int fd)
- static FILE * **funcompress** (const char *path)
Open a pipe to uncompress the specified file.
- static FILE * **fcompress** (const char *path)
Open a pipe to compress the specified file.
- FILE * **fopen_gen** (const char *path, const char *mode)
Generalized fopen function. fopen_gen is to be used as fopen. Can be used in read and in write mode. When used in read mode with a compressed extension, the file will be first decompressed and then read. When used in write mode with a compressed extension, the output will be compressed.

4.30.1 Detailed Description

Uncompress/compress input/output files using pipes.

Hook the standard file opening functions, open, fopen and fopen64. If the extension of the file being opened indicates the file is compressed (.gz, .bz2, .xz), when opening in the reading mode a pipe to a program is opened that decompresses that file (gunzip, bunzip2 or xzdec) and return a handle to the open pipe. When opening in the writing mode (only for .gz, .bam), a pipe to a program is opened that compresses the output.

Author

Paula Perez paulaperezrubio@gmail.com

Date

03.08.2017

Warning

vfork vs fork to be checked!

Note

- original copyright note - (reading mode, original C++ code) author: Shaun Jackman sjackman@bcgsc.ca, <https://github.com/bcgsc>, filename: Uncompress.cpp

4.30.2 Function Documentation**4.30.2.1 static int compress (const char * *path*) [static]**

Open a pipe to compress output. Open a pipe to uncompress the specified file. Not thread safe.

Returns

a file descriptor

4.30.2.2 static FILE* fcompress (const char * *path*) [static]

Open a pipe to compress the specified file.

Returns

a FILE pointer

4.30.2.3 FILE* fopen_gen (const char * *path*, const char * *mode*)

Generalized fopen function. fopen_gen is to be used as fopen. Can be used in read and in write mode. When used in read mode with a compressed extension, the file will be first decompressed and then read. When used in write mode with a compressed extension, the output will be compressed.

Returns

a FILE pointer

4.30.2.4 static FILE* funcompress (const char * *path*) [static]

Open a pipe to uncompress the specified file.

Returns

a FILE pointer

4.30.2.5 static int uncompress (const char * *path*) [static]

Open a pipe to uncompress file. Open a pipe to uncompress the specified file. Not thread safe.

Returns

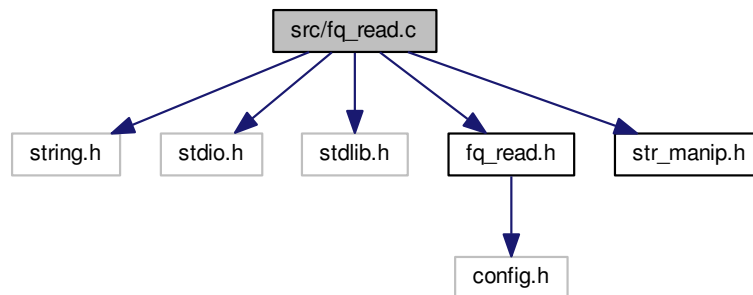
a file descriptor

4.31 src/fq_read.c File Reference

fastq entries manipulations (read/write)

```
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include "fq_read.h"
#include "str_manip.h"
```

Include dependency graph for fq_read.c:



Functions

- int [get_fqread](#) ([Fq_read](#) *seq, char *buffer, int pos1, int pos2, int nline, int read_len, int filter)
reads fastq line from a buffer
- int [string_seq](#) ([Fq_read](#) *seq, char *char_seq)
writes the fq entry in a string

4.31.1 Detailed Description

fastq entries manipulations (read/write)

Author

Paula Perez paulaperezrubio@gmail.com

Date

03.08.2017

4.31.2 Function Documentation

4.31.2.1 int [get_fqread](#) ([Fq_read](#) * seq, char * buffer, int pos1, int pos2, int nline, int read_len, int filter)

reads fastq line from a buffer

a fastq line is read from a buffer and the relevant information is stored in a structure **Fq_read**. Depending on the value of **filter**, information about whether the read was trimmed is stored.

Parameters

| | |
|-----------------|--|
| <i>seq</i> | pointer to Fq_read , where the info will be stored. |
| <i>buffer</i> | variable where the file being read is stored. |
| <i>pos1</i> | buffer start position of the line. |
| <i>pos2</i> | buffer end position of the line. |
| <i>nline</i> | file line number being read. |
| <i>read_len</i> | predefined read length |
| <i>filter</i> | 0 original file, 1 file filtered with filter_trim, 2 file filtered with another tool |

4.31.2.2 int string_seq (Fq_read * seq, char * char_seq)

writes the fq entry in a string

Parameters

| | |
|-----------------|--|
| <i>seq</i> | pointer to Fq_read , where the info will be stored. |
| <i>char_seq</i> | pointer to buffer, where the sequence will be stored |

Warning

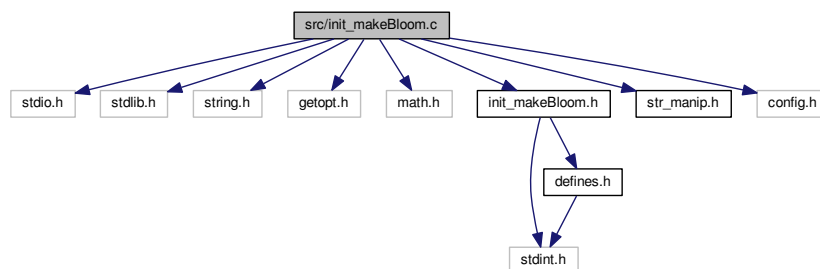
change the call to sprintf to snprintf

4.32 src/init_makeBloom.c File Reference

Help dialog for makeBloom and initialization of the command line arguments.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <getopt.h>
#include <math.h>
#include "init_makeBloom.h"
#include "str_manip.h"
#include "config.h"
```

Include dependency graph for init_makeBloom.c:



Functions

- void [printHelpDialog_makeBloom](#) ()
Function that prints makeBloom help dialog when called.
- void [getarg_makeBloom](#) (int argc, char **argv)

Reads in the arguments passed through the command line to makeBloom. and stores them in the global variable par_MB.

Variables

- [lparam_makeBloom par_MB](#)

4.32.1 Detailed Description

Help dialog for makeBloom and initialization of the command line arguments.

Author

Paula Perez paulaperezrubio@gmail.com

Date

05.09.2017

4.32.2 Variable Documentation

4.32.2.1 lparam_makeBloom par_MB

Input parameters of makeBloom

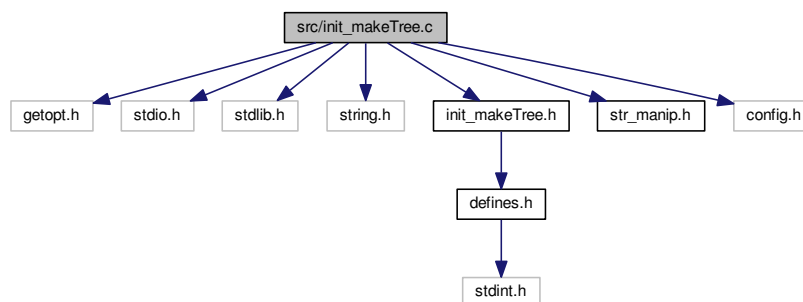
global variable: Input parameters of makeTree.

4.33 src/init_makeTree.c File Reference

Help dialog for makeTree and initialization of the command line arguments.

```
#include <getopt.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "init_makeTree.h"
#include "str_manip.h"
#include "config.h"
```

Include dependency graph for init_makeTree.c:



Functions

- void [printHelpDialog_makeTree](#) ()

Function that prints makeTree help dialog when called.

- void [getarg_makeTree](#) (int argc, char **argv)

Reads in the arguments passed through the command line to makeTree. and stores them in the global variable par_MT.

Variables

- [lparam_makeTree par_MT](#)

4.33.1 Detailed Description

Help dialog for makeTree and initialization of the command line arguments.

Author

Paula Perez paulaperezrubio@gmail.com

Date

23.08.2017

4.33.2 Variable Documentation

4.33.2.1 [lparam_makeTree par_MT](#)

Input parameters of makeTree

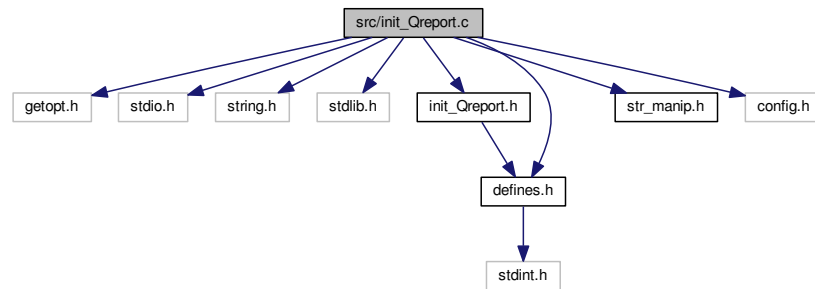
global variable: Input parameters of makeTree.

4.34 src/init_Qreport.c File Reference

Help dialog for Qreport and initialization of the command line arguments.

```
#include <getopt.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "init_Qreport.h"
#include "str_manip.h"
#include "config.h"
#include "defines.h"
```

Include dependency graph for `init_Qreport.c`:



Functions

- void [printHelpDialog_Qreport](#) ()
Function that prints Qreport help dialog when called.
- void [getarg_Qreport](#) (int argc, char **argv)
Reads in the arguments passed through the command line to Qreport. and stores them in the global variable `par_QR`.

Variables

- [lparam_Qreport par_QR](#)

4.34.1 Detailed Description

Help dialog for Qreport and initialization of the command line arguments.

Author

Paula Perez paulaperezrubio@gmail.com

Date

03.08.2017

4.34.2 Variable Documentation

4.34.2.1 [lparam_Qreport par_QR](#)

Input parameters of Qreport

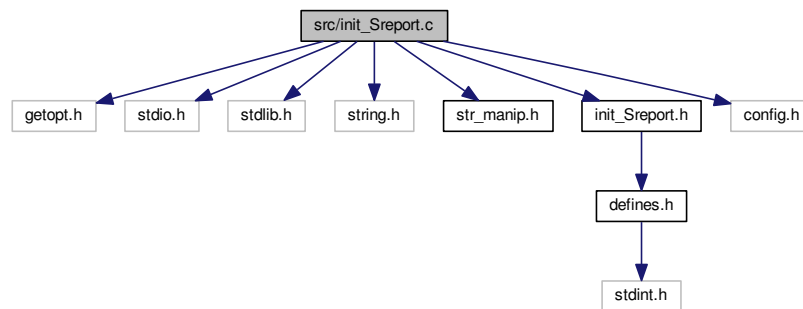
global variable: input parameters for Qreport

4.35 `src/init_Sreport.c` File Reference

Help dialog for Sreport and initialization of the command line arguments.

```
#include <getopt.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "str_manip.h"
#include "init_Sreport.h"
#include "config.h"
```

Include dependency graph for init_Sreport.c:



Functions

- void [printHelpDialog_Sreport](#) ()
Function that prints Sreport help dialog when called.
- void [getarg_Sreport](#) (int argc, char **argv)
Reads in the arguments passed through the command line to Sreport. and stores them in the global variable `par_SR`.

Variables

- [lparam_Sreport](#) `par_SR`

4.35.1 Detailed Description

Help dialog for Sreport and initialization of the command line arguments.

Author

Paula Perez paulaperezrubio@gmail.com

Date

09.08.2017

4.35.2 Variable Documentation

4.35.2.1 `lparam_Sreport` `par_SR`

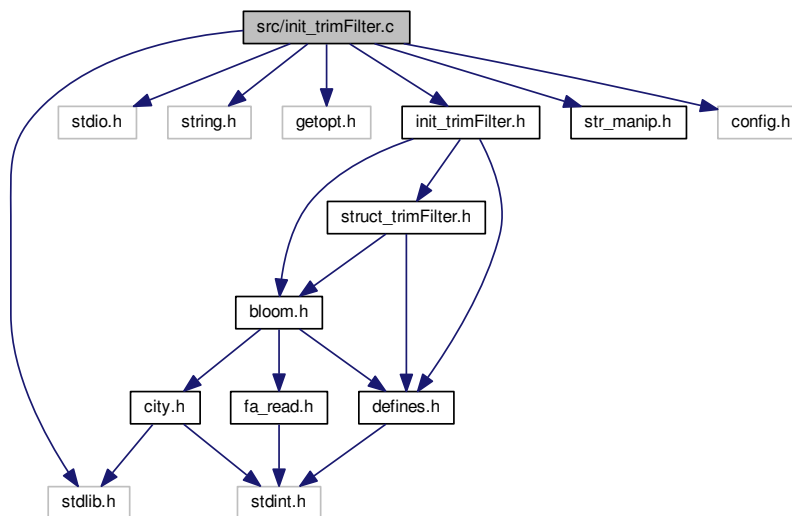
input parameters Sreport

4.36 src/init_trimFilter.c File Reference

help dialog for trimFilter and initialization of the command line arguments.

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <getopt.h>
#include "init_trimFilter.h"
#include "str_manip.h"
#include "config.h"
```

Include dependency graph for init_trimFilter.c:



Functions

- void [printHelpDialog_trimFilter](#) ()
Function that prints trimFilter help dialog when called.
- void [getarg_trimFilter](#) (int argc, char **argv)
Reads in the arguments passed through the command line to trimFilter. and stores them in the global variable par_TF.

Variables

- [lparam_trimFilter](#) par_TF

4.36.1 Detailed Description

help dialog for trimFilter and initialization of the command line arguments.

Author

Paula Perez paulaperezrubio@gmail.com

Date

24.08.2017

4.36.2 Variable Documentation

4.36.2.1 Iparam_trimFilter par_TF

Input parameters of makeTree

global variable: Input parameters trimFilter.

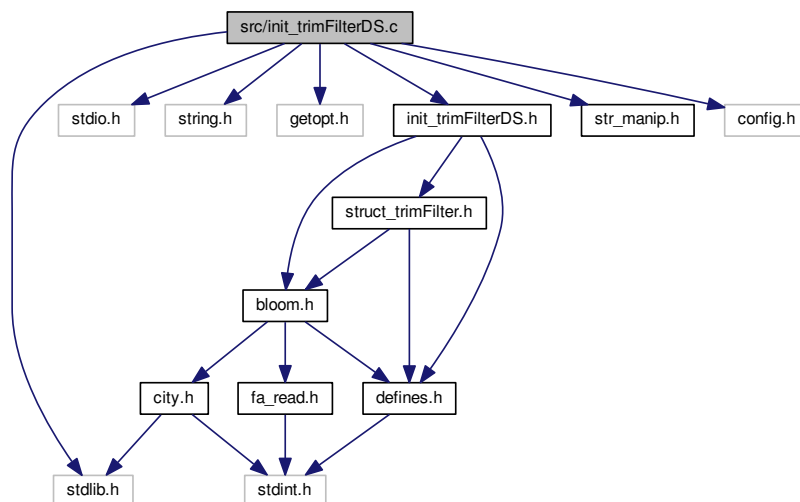
global variable: Input parameters of makeTree.

4.37 src/init_trimFilterDS.c File Reference

help dialog for trimFilterDS and initialization of the command line arguments.

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <getopt.h>
#include "init_trimFilterDS.h"
#include "str_manip.h"
#include "config.h"
```

Include dependency graph for init_trimFilterDS.c:



Functions

- void [printHelpDialog_trimFilterDS](#) ()
Function that prints trimFilterDS help dialog when called.
- void [getarg_trimFilterDS](#) (int argc, char **argv)
Reads in the arguments passed through the command line to trimFilterDS and stores them in the global variable par_TF.

Variables

- [lparam_trimFilter par_TF](#)

4.37.1 Detailed Description

help dialog for trimFilterDS and initialization of the command line arguments.

Author

Paula Perez paulaperezrubio@gmail.com

Date

07.10.2017

4.37.2 Variable Documentation

4.37.2.1 [lparam_trimFilter par_TF](#)

Input parameters of makeTree

global variable: Input parameters trimFilter.

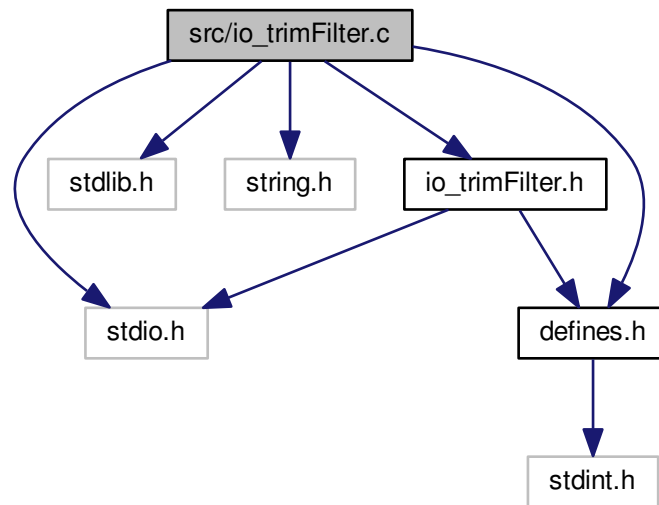
global variable: Input parameters of makeTree.

4.38 [src/io_trimFilter.c](#) File Reference

buffer fq output, write summary file

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "io_trimFilter.h"
#include "defines.h"
```

Include dependency graph for io_trimFilter.c:



Functions

- void `buffer_output` (FILE *fout, const char *str, const int len, const int fd_i)
buffers the output before writing to disk, writes out summary
- void `write_summary_TF` (Stats_TF tf_stats, char *filename)
writes stats of filtering to summary file (binary)

4.38.1 Detailed Description

buffer fq output, write summary file

Author

Paula Perez paulaperezrubio@gmail.com

Date

29.08.2017

4.38.2 Function Documentation

4.38.2.1 void `buffer_output` (FILE * *fout*, const char * *str*, const int *len*, const int *fd_i*)

buffers the output before writing to disk, writes out summary

Parameters

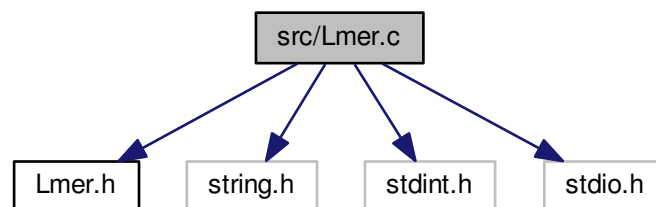
| | |
|-------------|--|
| <i>fout</i> | FILE pointer where we might write to disk; |
| <i>str</i> | string we want to add |
| <i>len</i> | length of the string we want to add |
| <i>fd_i</i> | identifier: GOOD, ADAP, CONT, LOWQ, NNNN |

4.39 src/Lmer.c File Reference

Manipulation of Lmers and sequences.

```
#include "Lmer.h"
#include <string.h>
#include <stdint.h>
#include <stdio.h>
```

Include dependency graph for Lmer.c:



Functions

- void [init_map](#) ()
Initialize lookup table fw_1B.
- void [init_map_SA](#) ()
Initialize lookup table fw_1B (for SA)
- void [Lmer_sLmer](#) (char *Lmer, int L)
Transforms an Lmer to the convention stored in the lookup table fw_1B.
- void [rev_comp](#) (char *sLmer, int L)
Obtains the reverse complement, for {"000","001","002","003"}.
- void [rev_comp2](#) (char *sLmer, int L)
Obtains the reverse complement, for {"001","002","003","004"}.

Variables

- uint8_t [fw_1B](#) [256]
- uint8_t [bw_1B](#) [256]
- uint8_t [Nencode](#)

4.39.1 Detailed Description

Manipulation of Lmers and sequences.

Author

Paula Perez paulaperezrubio@gmail.com

Date

18.08.2017

4.39.2 Function Documentation

4.39.2.1 void init_map ()

Initialize lookup table fw_1B.

{'a','c','g','t'} -> {'\000','\001','\002','\003'}, rest '\004'.

4.39.2.2 void init_map_SA ()

Initialize lookup table fw_1B (for SA)

{'a','c','g','t'} -> {'\001','\002','\003','\004'}, rest '\005'.

4.39.3 Variable Documentation

4.39.3.1 uint8_t bw_1B[256]

global variable. Lookup table.

4.39.3.2 uint8_t fw_1B[256]

global variable. Lookup table.

4.39.3.3 uint8_t Nencode

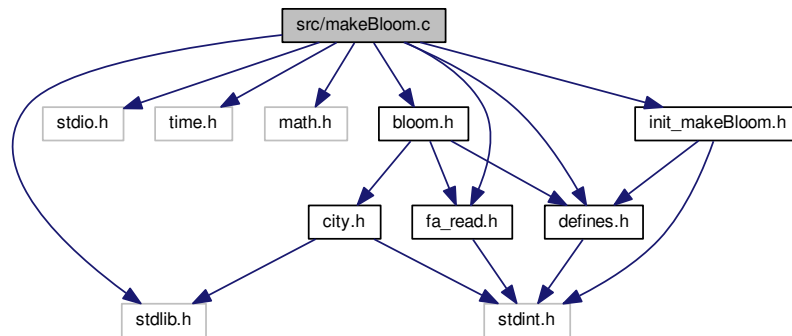
global variable. Encoding for N's(\004, or \005)

4.40 src/makeBloom.c File Reference

makeBloom main function

```
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
#include <math.h>
#include "defines.h"
#include "fa_read.h"
#include "bloom.h"
#include "init_makeBloom.h"
```

Include dependency graph for makeBloom.c:



Functions

- int `main` (int argc, char *argv[])
makeTree main function

Variables

- uint64_t `alloc_mem` = 0
- `lparam_makeBloom` par_MB

4.40.1 Detailed Description

makeBloom main function

Author

Paula Perez paulaperezrubio@gmail.com

Date

05.09.2017 This file contains the makeBloom main function. It reads a fasta file, constructs a bloom filter and stores it in a file. See README_makeTree.md for more details.

4.40.2 Variable Documentation

4.40.2.1 uint64_t alloc_mem = 0

global variable. Memory allocated in the heap.

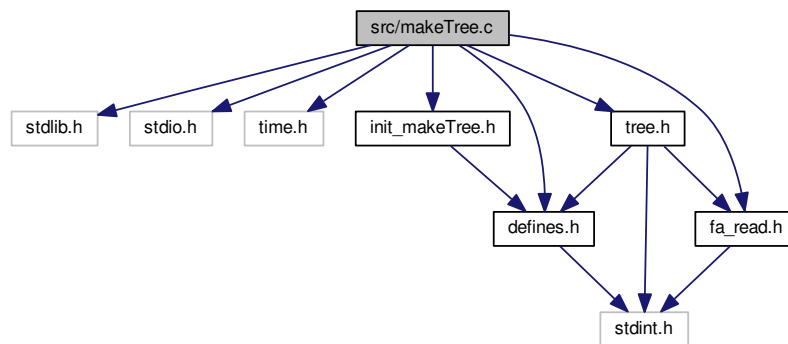
4.40.2.2 lparam_makeBloom par_MB

global variable: Input parameters of makeTree.

4.41 src/makeTree.c File Reference

makeTree main function

```
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
#include "defines.h"
#include "fa_read.h"
#include "tree.h"
#include "init_makeTree.h"
Include dependency graph for makeTree.c:
```



Functions

- int [main](#) (int argc, char *argv[])
makeTree main function

Variables

- uint64_t [alloc_mem](#) = 0
- [lparam_makeTree](#) [par_MT](#)

4.41.1 Detailed Description

makeTree main function

Author

Paula Perez paulaperezrubio@gmail.com

Date

23.08.2017 This file contains the makeTree main function. It reads a fasta file, constructs a 4-tree of depth L and stores it compressed in a file. See README_makeTree.md for more details.

4.41.2 Variable Documentation

4.41.2.1 uint64_t alloc_mem = 0

global variable. Memory allocated in the heap.

4.41.2.2 lparam_makeTree par_MT

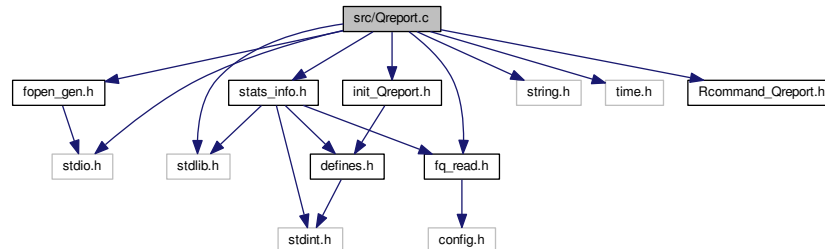
global variable: Input parameters of makeTree.

4.42 src/Qreport.c File Reference

QReport main function.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include "init_Qreport.h"
#include "fopen_gen.h"
#include "fq_read.h"
#include "stats_info.h"
#include "Rcommand_Qreport.h"
```

Include dependency graph for Qreport.c:



Functions

- int [main](#) (int argc, char *argv[])
Qreport main function.

Variables

- [lparam_Qreport](#) [par_QR](#)

4.42.1 Detailed Description

QReport main function.

Author

Paula Perez paulaperezrubio@gmail.com

Date

03.08.2017 This file contains the quality report main function. It reads a fastq file and creates a html quality report. See README_Qreport.md for more details.

4.42.2 Variable Documentation**4.42.2.1 lparam_Qreport par_QR**

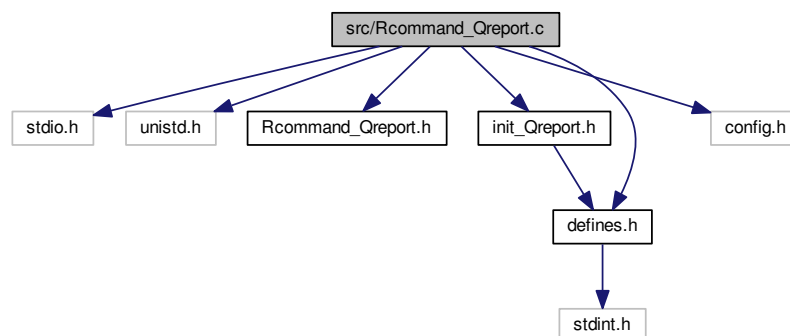
global variable: input parameters for Qreport

4.43 src/Rcommand_Qreport.c File Reference

get Rscript command for Qreport

```
#include <stdio.h>
#include <unistd.h>
#include "Rcommand_Qreport.h"
#include "init_Qreport.h"
#include "defines.h"
#include "config.h"
```

Include dependency graph for Rcommand_Qreport.c:

**Functions**

- `char * command_Qreport ()`
returns Rscript command that generates the quality report in html

Variables

- `lparam_Qreport par_QR`

4.43.1 Detailed Description

get Rscript command for Qreport

Author

Paula Perez paulaperezrubio@gmail.com

Date

07.08.2017

4.43.2 Variable Documentation

4.43.2.1 lparam_Qreport par_QR

input parameters Qreport

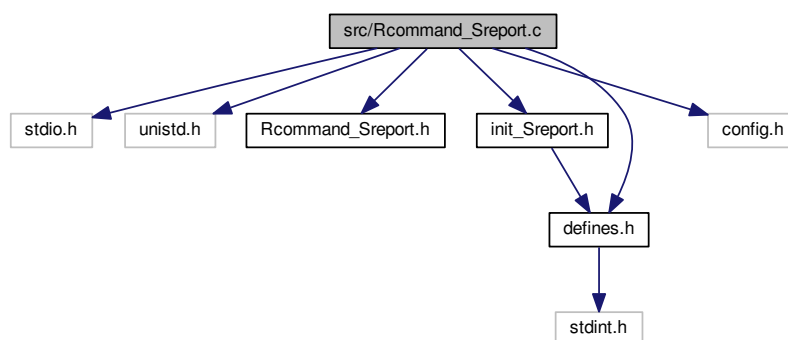
global variable: input parameters for Qreport

4.44 src/Rcommand_Sreport.c File Reference

get Rscript command for Sreport

```
#include <stdio.h>
#include <unistd.h>
#include "Rcommand_Sreport.h"
#include "init_Sreport.h"
#include "defines.h"
#include "config.h"
```

Include dependency graph for Rcommand_Sreport.c:



Functions

- `char * command_Sreport ()`
returns Rscript command that generates the summary report in html

Variables

- `lparam_Sreport par_SR`

4.44.1 Detailed Description

get Rscript command for Sreport

Author

Paula Perez paulaperezrubio@gmail.com

Date

09.08.2017

4.44.2 Function Documentation

4.44.2.1 char* command_Sreport()

returns Rscript command that generates the summary report in html

```
1 # To run between quotation marks after: Rscript_RBioC -e (Rscript)
2 inputfolder = normalizePath( <par.SR.inputfolder>, mustWork = TRUE);
3 output = <par_SR.outputfile>;
4 output_file = gsub('.* /', '', output);
5 path = gsub('[^/]+$' , '', output);
6 if (path != '') {
7   outputfile = paste0(normalizePath(path, mustWork = TRUE), '/', outputfile);
8 } else {
9   outputfile = paste0(cwd, '/', output_file); # cwd: current working dir
10 };
11 rmarkdown::render(<par_SR.Rmd_file>,
12                  params = list(inputfolder = inputfolder, version= VERSION),
13                  output_file = output_file)
```

4.44.3 Variable Documentation

4.44.3.1 lparam_Sreport par_SR

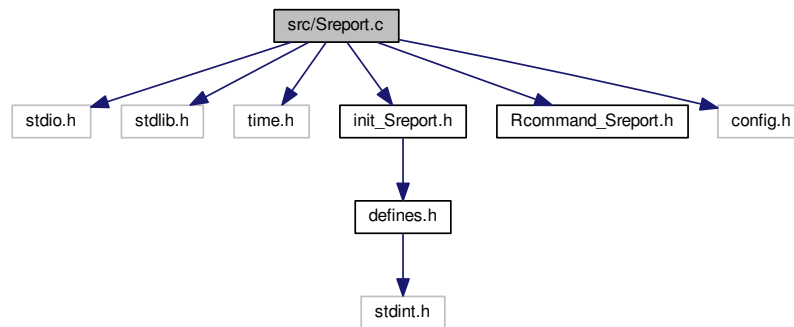
input parameters Sreport

4.45 src/Sreport.c File Reference

Sreport main function.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "init_Sreport.h"
#include "Rcommand_Sreport.h"
#include "config.h"
```

Include dependency graph for Sreport.c:



Functions

- `int main (int argc, char *argv[])`
Qreport main function.

Variables

- `lparam_Sreport par_SR`

4.45.1 Detailed Description

Sreport main function.

Author

Paula Perez paulaperezrubio@gmail.com

Date

09.08.2017 This file contains the summary report main function. Given a folder containing `*bin` as from Qreport output, Sreport generates a summary report in html format. See `README_Sreport.md` for more details.

4.45.2 Variable Documentation

4.45.2.1 lparam_Sreport par_SR

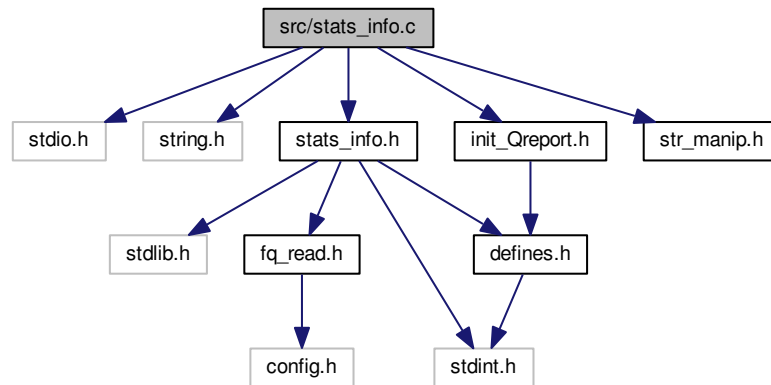
input parameters Sreport

4.46 src/stats_info.c File Reference

Construct the quality report variables and update them.

```
#include <stdio.h>
#include <string.h>
#include "stats_info.h"
#include "init_Qreport.h"
#include "str_manip.h"
```

Include dependency graph for stats_info.c:



Functions

- void [get_tile_lane](#) (char *line1, int *tile, int *lane)
get tile number from first line in fastq entry.
- static int [belongsto](#) (int k, int *qual_tags, int nQ)
returns 1 if k is in qual_tags, 0 otherwise.
- static int [cmpfunc](#) (const void *a, const void *b)
comparison function for qsort
- void [init_info](#) (Info *res)
Initialization of a Info type.
- void [free_info](#) (Info *res)
frees allocated memory in Info
- void [read_info](#) (Info *res, char *file)
Read Info from binary file.
- void [write_info](#) (Info *res, char *file)
Write info to binary file.
- void [print_info](#) (Info *res, char *infofile)
print Info to a textfile
- void [get_first_tile](#) (Info *res, Fq_read *seq)
gets first tile
- void [update_info](#) (Info *res, Fq_read *seq)
updates Info with Fq_read
- int [update_ACGT_counts](#) (uint64_t *ACGT_low, char ACGT)
update, for current tile, ACGT counts.
- void [update_QPosTile_table](#) (Info *res, Fq_read *seq)
update QPostile table
- void [update_ACGT_pos](#) (uint64_t *ACGT_pos, Fq_read *seq, int read_len)

- *update ACGT_pos*
- void [resize_info](#) ([Info](#) *res)
- *resize Info*

Variables

- [lparam_Qreport](#) [par_QR](#)

4.46.1 Detailed Description

Construct the quality report variables and update them.

Author

Paula Perez paulaperezrubio@gmail.com

Date

04.08.2017

4.46.2 Function Documentation

4.46.2.1 void [get_tile_lane](#) (char * *line1*, int * *tile*, int * *lane*)

get tile number from first line in fastq entry.

Parameters

| | |
|--------------|------------------------------------|
| <i>line1</i> | first line of a fastq entry |
| <i>tile</i> | int* where the tile will be stored |
| <i>lane</i> | int* where the lane will be stored |

See also

http://wiki.christophchamp.com/index.php?title=FASTQ_format

Only Illumina sequence identifiers are allowed. The line is inspected, and the number of ':' is obtained. The function exits with an error if the number of semicolons is different from 4 or 9.

4.46.2.2 void [init_info](#) ([Info](#) * *res*)

Initialization of a [Info](#) type.

It sets: nQ, read_len, ntiles, minQ and the dimensions of the arrays. Initializes the rest of the variables to zero and allocates memory to the arrays initializing them to 0 (calloc).

4.46.2.3 void [resize_info](#) ([Info](#) * *res*)

resize [Info](#)

At the end of the program, resize the structure [Info](#), and adapt it to the actual number of tiles and the actual number of different quality values present.

4.46.2.4 int update_ACGT_counts (uint64_t * ACGT_low, char ACGT)

update, for current tile, ACGT counts.

Makes update of ACGT counts for the current tile. Can be used with variables: lowQ_ACGT_tile and ACGT_tile

4.46.3 Variable Documentation

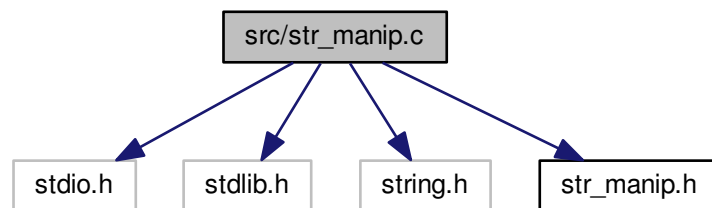
4.46.3.1 lparam_Qreport par_QR

global variable: input parameters for Qreport

4.47 src/str_manip.c File Reference

functions that do string manipulation

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "str_manip.h"
Include dependency graph for str_manip.c:
```



Macros

- `#define __isascii_c(c) (((c) & ~0x7f) == 0)`

Functions

- int `str_isascii` (char *s)
return nonzero iff all elements in the string are in the ASCII set.
- int `strindex` (char *s, char *t)
returns index of t in s (start, first occurrence)
- int `count_char` (char *str, char sep)
returns the # of occurrences of char c in string s
- int `strindexC` (char *s, char sep)
returns index of t in s (start, first occurrence)
- `Split strsplit` (char *str, char sep)
Separates strings by a separator.

4.47.1 Detailed Description

functions that do string manipulation

Author

Paula Perez paulaperezrubio@gmail.com

Date

03.08.2017

4.47.2 Macro Definition Documentation

4.47.2.1 `#define __isascii_c(c) (((c) & ~0x7f) == 0)`

If C is a 7 bit value.

4.47.3 Function Documentation

4.47.3.1 `int strindex (char * s, char * t)`

returns index of t in s (start, first occurrence)

Parameters

| | |
|----------|-----------------------------|
| <i>s</i> | string to be checked. |
| <i>t</i> | substring to be found in s. |

4.47.3.2 `int strindexC (char * s, char sep)`

returns index of t in s (start, first occurrence)

Parameters

| | |
|------------|-----------------------|
| <i>s</i> | string to be checked. |
| <i>sep</i> | char, separator |

4.47.3.3 `Split strsplit (char * str, char sep)`

Separates strings by a separator.

Parameters

| | |
|------------|------------------|
| <i>str</i> | input string |
| <i>sep</i> | separator (char) |

Returns

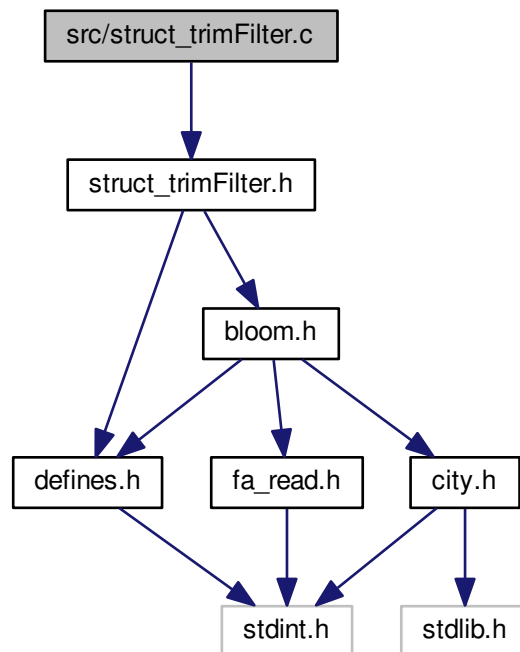
array of strings containing the substrings in the input separated

4.48 `src/struct_trimFilter.c` File Reference

function that frees the memory of parTF (structure storing the trimFilter/trimFilterDS input arguments).


```
#include "struct_trimFilter.h"
```

Include dependency graph for struct_trimFilter.c:



Functions

- void `free_parTF` (`lparam_trimFilter *ptr_parTF`)
frees the allocated memory in `lparam_trimFilter`

4.48.1 Detailed Description

function that frees the memory of parTF (structure storing the trimFilter/trimFilterDS input arguments).

Author

Paula Perez paulaperezrubio@gmail.com

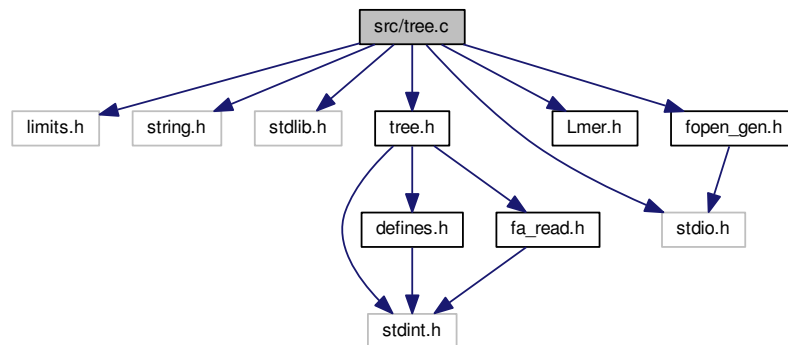
Date

07.10.2017

4.49 src/tree.c File Reference

Construction of tree, check paths, write tree, read in tree.

```
#include <limits.h>
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include "tree.h"
#include "Lmer.h"
#include "fopen_gen.h"
Include dependency graph for tree.c:
```



Functions

- `Node * get_new_pool (Tree *tree_ptr)`
reallocs pool_2D (++NPOOL_2D) if all existing nodes have been used
- `Node * new_node_buf (Tree *tree_ptr)`
moves to the next node (allocating new memory if necessary)
- `void free_all_nodes (Tree *tree_ptr)`
frees the whole tree structure
- `void insert_Lmer (Tree *tree_ptr, char *Lmer)`
Lmer insertion in the tree (depth L).
- `void insert_entry (Tree *tree_ptr, Fa_entry *entry)`
fasta entry insertion in the tree (depth L).
- `Tree * tree_from_fasta (Fa_data *fasta, int L)`
create Tree structure from fasta structure.
- `double check_path (Tree *tree_ptr, char *read, int Lread)`
checks if read is found in tree and outputs a score
- `void save_tree (Tree *tree_ptr, char *filename)`
saves Tree to disk in filename
- `Tree * read_tree (char *filename)`
read tree from file

Variables

- `uint64_t alloc_mem`

4.49.1 Detailed Description

Construction of tree, check paths, write tree, read in tree.

Author

Paula Perez paulaperezrubio@gmail.com

Date

23.08.2017

4.49.2 Function Documentation

4.49.2.1 double check_path (Tree * tree_ptr, char * read, int Lread)

checks if read is found in tree and outputs a score

Parameters

| | |
|-----------------|----------------------------|
| <i>tree_ptr</i> | pointer to Tree structure |
| <i>read</i> | Read or reverse complement |
| <i>Lread</i> | length of read |

Returns

score = (number of Lmers of reads found in read) / (Lread-L+1)

4.49.2.2 void free_all_nodes (Tree * tree_ptr)

frees the whole tree structure

Parameters

| | |
|-----------------|---------------------------|
| <i>tree_ptr</i> | pointer to Tree structure |
|-----------------|---------------------------|

This function deallocates the memory allocated in a Tree structure.

4.49.2.3 Node* get_new_pool (Tree * tree_ptr)

reallocs pool_2D (++NPOOL_2D) if all existing nodes have been used

Parameters

| | |
|-----------------|---------------------------|
| <i>tree_ptr</i> | pointer to Tree structure |
|-----------------|---------------------------|

4.49.2.4 Node* new_node_buf (Tree * tree_ptr)

moves to the next node (allocating new memory if necessary)

Parameters

| | |
|-----------------|---------------------------|
| <i>tree_ptr</i> | pointer to Tree structure |
|-----------------|---------------------------|

Returns

address to next node

The function checks if there are available nodes (information stored in the variable `tree_ptr -> pool_available`) and goes to the next node. If there is no nodes left, it allocates a new `pool_1D`, and if there is no room left in the outer dimension, it reallocates `NPOOL_2D` more `Node*`'s. If the number of nodes reaches `UINT_MAX`, the program returns an error message and exits.

4.49.2.5 Tree* read_tree (char * filename)

read tree from file

Parameters

| | |
|-----------------|--------------------------|
| <i>filename</i> | string with the filename |
|-----------------|--------------------------|

Returns

pointer to Tree structure

This function unwinds the process carried out in `save_tree` and assigns addresses to the children of every given node.

4.49.2.6 void save_tree (Tree * tree_ptr, char * filename)

saves Tree to disk in filename

Parameters

| | |
|-----------------|----------------------------|
| <i>tree_ptr</i> | pointer to Tree structure |
| <i>filename</i> | string containing filename |

The tree structure is stored as follows: every address is stored in a `uint32_t` (we are not allowing trees with more than `UINT_MAX` nodes). For every node, the addresses of the children are stored in the following fashion:

- If it is pointing to NULL: 0.
- Otherwise: `i2`, the index in the outer dimension of `pool_2D` is identified, and the difference `jump = pool_2D[i2].children[k] - pool_2D[i2]` is computed. `i2*NPOOL_D1 + jump` is then stored for child `k`.

4.49.2.7 Tree* tree_from_fasta (Fa_data * fasta, int L)

create Tree structure from fasta structure.

Parameters

| | |
|--------------|----------------------------|
| <i>fasta</i> | pointer to fasta structure |
| <i>L</i> | tree length |

4.49.3 Variable Documentation**4.49.3.1 uint64_t alloc_mem**

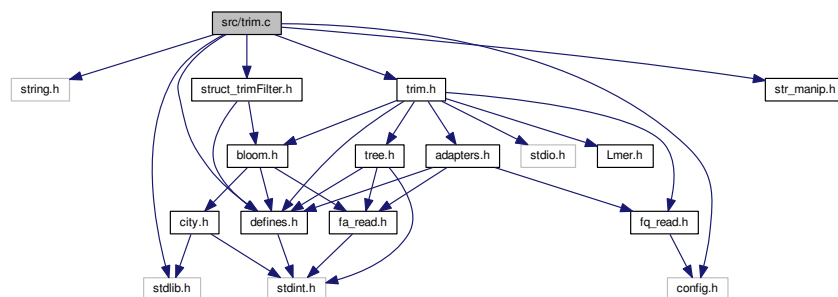
global variable. Memory allocated in the heap.

4.50 src/trim.c File Reference

trims/filter sequences after Quality, N's contaminations.

```
#include <string.h>
#include <stdlib.h>
#include "trim.h"
#include "str_manip.h"
#include "defines.h"
#include "config.h"
#include "struct_trimFilter.h"
```

Include dependency graph for trim.c:



Macros

- `#define TRIM_STRING 20`

Functions

- static int `no_N` (`Fq_read *seq`)
checks if a sequence contains any non standard base callings (N's)
- static int `Nfree_Lmer` (`Fq_read *seq`, int minL)
Finds the largest Nfree sub-seq and keeps it if larger than minL.
- static int `Ntrim_ends` (`Fq_read *seq`, int minL)
trims a read if N's are at the ends and the remaining sub-seq \geq minL
- static int `no_lowQ` (`Fq_read *seq`, int minQ)
checks if a sequence contains lowQ nucleotides
- static int `Qtrim_ends` (`Fq_read *seq`, int minQ, int minL)
trims a read if lowQs are at the ends and remaining sub-seq \geq minL
- static int `Qtrim_frac` (`Fq_read *seq`, int minQ, int nlowQ)
accepts the sequence as is if there are less than nlowQ
- static int `Qtrim_endsfrac` (`Fq_read *seq`, int minQ, int minL, int nlowQ)
trims the ends for lowQ. The rest is kept if it contains $<$ nlowQ lowQ
- int `Qtrim_global` (`Fq_read *seq`, int left, int right, char type)
trims left from the left and right from the right
- static int `align_uint32` (`Fq_read *seq`, `Ad_seq *ptr_adap`, bool all)
alignment search between a fq read, and an adapter sequence, with a seed of 8 nucleotides.
- static int `align_uint64` (`Fq_read *seq`, `Ad_seq *ptr_adap`)
Alignment search between a fq read, and an adapter sequence, w with a seed of 8 nucleotides.
- int `trim_adapter` (`Fq_read *seq`, `Ad_seq *adap_list`)

- trims sequence based on presence of N nucleotides*
 - int `trim_sequenceN` (`Fq_read` *seq)
- trims sequence based on presence of N nucleotides*
 - int `trim_sequenceQ` (`Fq_read` *seq)
- trims sequence based on lowQ base callings*
 - bool `is_read_inTree` (`Tree` *tree_ptr, `Fq_read` *seq)
- check if Lread is contained in tree. It computes the score for the read and its reverse complement; if one of them exceeds the user selected threshold, it returns true. Otherwise, it returns false.*
 - bool `is_read_inBloom` (`Bfilter` *ptr_bf, `Fq_read` *seq, `Bfkmer` *ptr_bfkmer)
- checks if a read is in Bloom filter. It computes the score for the read and returns true if it exceeds the user selected threshold. Returns false otherwise.*

Variables

- int `Nencode`
- `lparam_trimFilter` `par_TF`

4.50.1 Detailed Description

trims/filter sequences after Quality, N's contaminations.

Author

Paula Perez paulaperezrubio@gmail.com

Date

24.08.2017

4.50.2 Macro Definition Documentation

4.50.2.1 #define TRIM_STRING 20

maximal length of trimming info string.

4.50.3 Function Documentation

4.50.3.1 static int align_uint32 (`Fq_read` * seq, `Ad_seq` * ptr_adap, bool all) [static]

alignment search between a fq read, and an adapter sequence, with a seed of 8 nucleotides.

This function checks whether there is adapter contamination in a given read. It works stand alone if the adapter is shorter than 16 nucleotides, and is called from `align_uint64` when no 16-nucleotides long seeds are found. The criteria are the same as in `align_uint64`, the seed length being 8-nucleotides long instead of 16. See the `align_uint64` documentation for more details.

Parameters

| | |
|-----------------------|---------------------------------|
| <code>seq</code> | pointer to <code>Fq_read</code> |
| <code>ptr_adap</code> | pointer to <code>Ad_seq</code> |

| | |
|------------|--|
| <i>all</i> | true if the whole read has to be swept, false if only the ends. When this function is called from align_uint64, only the ends need to be considered. |
|------------|--|

Returns

-1 error, 0 discarded, 1 accepted as is, 2 accepted and trimmed

Note

Global input parameters from par_TF are also used

See also

[Adapter](#)
[lparam_trimFilter](#)
[align_uint64](#)
[pack_adapter](#)
[obtain_score](#)

4.50.3.2 static int align_uint64 (Fq_read * seq, Ad_seq * ptr_adap) [static]

Alignment search between a fq read, and an adapter sequence, w with a seed of 8 nucleotides.

Parameters

| | |
|-----------------|---------------------------|
| <i>seq</i> | pointer to Fq_read |
| <i>ptr_adap</i> | pointer to Ad_seq |

Returns

-1 error, 0 discarded, 1 accepted as is, 2 accepted and trimmed

Note

Global input parameters from par_TF are used

See also

[Adapter](#)
[lparam_trimFilter](#)
[align_uint32](#)
[pack_adapter](#)
[obtain_score](#)

This function checks whether there is adapter contamination in a given read. We start by looking for 16-nucleotides long seeds, where a user defined number of mismatches is allowed. If found, a score is computed. If the score is larger than the user defined threshold and the number of matched nucleotides exceeds MIN_NMATCHES (12), then the read is trimmed if the remaining part is longer than minL (user defined) and discarded otherwise. If no 16-nucleotides long seeds are found, we proceed with 8-nucleotides long seeds (see [align_uint32](#)) and apply the same criteria to trim/discard a read. A list of possible situations follows, to illustrate how it works (minL=25, mismatches=2):

```

1 ADAPTER: CAAGCAGAAGACGGCATACGAG
2 REV_COM: AGATCGGAAGAGCTCGTATGCC
3
4 CASE1A:  CACAGTCGATCAGCGAGCAGGCATTTCATGCTGAGATCGGAAGAGATCGTATG
5           |||||X|||----
6           AGATCGGAAGAGCTCGTATG

```

```

7         - Seed: 16 Nucleotides
8         - Return: 2, TRIMA:0:31
9 CASE1B: CACATCATCGCTAGCTATCGATCGATCGATGCTATGCAAGATCGGAAGAGCT
10                                     |||||-----
11                                     AGATCGGAAGAGCT
12         - Seed: 8 Nucleotides
13         - Return: 2, TRIMA:0:37
14 CASE1C: CACATCATCGCTAGCTATCGATCGATCGATGCTATGCACGAAGATCGGAAGA
15                                     |||||----
16                                     AGATCGGAAGA
17         - Seed: 8 Nucleotides
18         - Return: 1, reason: Match length < 12
19 CASE2A: CATACTACGAGCTAGCTAGAGATCGGAAGAGCTCGTATGCCAGCATCGA
20                                     |||||-----
21                                     AGATCGGAAGAGCTCGTATGCC
22         - Seed: 16 Nucleotides
23         - Return: 0, reason: remaining read too short.
24 CASE2B: CCACAGTACAATACATCACGAGCTAGCTAGAGATCGGAAGAGCTCGTATGCC
25                                     |||||-----
26                                     AGATCGGAAGAGCTCGTATGCC
27         - Seed: 16 Nucleotides
28         - Return: 2, TRIMA:0:28
29 CASE3A: TATGCCGTCTTCTGCTTGCGAGTGCATGCTGATGCATGCTGCATGCTAGCTGC
30 |||||-----
31 TATGCCGTCTTCTGCTTG
32         - Seed: 16 Nucleotides
33         - Return: 0, reason: remaining read too short
34 CASE3B: CGTCTTCTGCTTGCCGATCGATGCTAGCTACGATCGTCGAGCTAGCTACGTG
35 |||||-----
36 CGTCTTCTGCTTG
37         - Seed: 8 Nucleotides
38         - Return: 0, reason: remaining read too short
39 CASE3C: TCTTCTGCTTGCCGATCGATGCTAGCTACGATCGTCGAGCTAGCTACGTGCG
40 |||||----
41 TCTTCTGCTTG
42         - Seed: 8 Nucleotides
43         - Return: 1, reason: Match length < 12

```

4.50.3.3 bool is_read_inBloom (Bfilter * ptr_bf, Fq_read * seq, Bfkmer * ptr_bfkmer)

checks if a read is in Bloom filter. It computes the score for the read and returns true if it exceeds the user selected threshold. Returns false otherwise.

Parameters

| | |
|-------------------|---|
| <i>ptr_bf</i> | pointer to Bfilter |
| <i>seq</i> | fastq read |
| <i>ptr_bfkmer</i> | pointer to Procs_kmer structure (will store global) |

Returns

true if read was found, false otherwise

4.50.3.4 bool is_read_inTree (Tree * tree_ptr, Fq_read * seq)

check if Lread is contained in tree. It computes the score for the read and its reverse complement; if one of them exceeds the user selected threshold, it returns true. Otherwise, it returns false.

Parameters

| | |
|-----------------|---------------------------|
| <i>tree_ptr</i> | pointer to Tree structure |
| <i>seq</i> | fastq read |

Returns

true if read was found, false otherwise

4.50.3.5 `static int Nfree_Lmer (Fq_read * seq, int minL) [static]`

Finds the largest Nfree sub-seq and keeps it if larger than minL.

Parameters

| | |
|-------------|---------------------------------|
| <i>seq</i> | fastq read |
| <i>minL</i> | minimum accepted trimmed length |

Returns

0 if not used, 1 if accepted as is, 2 if accepted and trimmed

4.50.3.6 `static int no_lowQ (Fq_read * seq, int minQ) [static]`

checks if a sequence contains lowQ nucleotides

Parameters

| | |
|-------------|--------------------------------|
| <i>seq</i> | fastq read |
| <i>minQ</i> | minimum accepted quality value |

Returns

0 if seq contains lowQ nucleotides, 1 otherwise

4.50.3.7 `static int no_N (Fq_read * seq) [static]`

checks if a sequence contains any non standard base callings (N's)

Returns

0 if no N's found, 1 if N's found

This function checks if any of the base callings in a given fastq read is different from A, C, G, T. Basically, any char different from the former ones is classified as N.

4.50.3.8 `static int Ntrim_ends (Fq_read * seq, int minL) [static]`

trims a read if N's are at the ends and the remaining sub-seq >= minL

Parameters

| | |
|-------------|---------------------------------|
| <i>seq</i> | fastq read |
| <i>minL</i> | minimum accepted trimmed length |

Returns

0 if not used, 1 no N's found, 2 if accepted and trimmed

4.50.3.9 `static int Qtrim_ends (Fq_read * seq, int minQ, int minL) [static]`

trims a read if lowQs are at the ends and remaining sub-seq >= minL

Parameters

| | |
|-------------|---------------------------------|
| <i>seq</i> | fastq read |
| <i>minQ</i> | minimum accepted quality value |
| <i>minL</i> | minimum accepted trimmed length |

Returns

0 if not used, 1 if accepted as is, 2 if accepted and trimmed

4.50.3.10 `static int Qtrim_endsfrac (Fq_read * seq, int minQ, int minL, int nlowQ) [static]`

trims the ends for lowQ. The rest is kept if it contains < nlowQ lowQ

Parameters

| | |
|--------------|--|
| <i>seq</i> | fastq read |
| <i>minQ</i> | minimum accepted quality value |
| <i>minL</i> | minimum accepted trimmed length |
| <i>nlowQ</i> | threshold on lowQ nucleotides (>= NOT allowed) |

Returns

0 if not used, 1 if accepted as is

Trims the ends if there are lowQ bases. From the remaining part, it counts how many lowQ bases there are and keeps it if there are less than nlowQ.

4.50.3.11 `static int Qtrim_frac (Fq_read * seq, int minQ, int nlowQ) [static]`

accepts the sequence as is if there are less than nlowQ

Parameters

| | |
|--------------|--|
| <i>seq</i> | fastq read |
| <i>minQ</i> | minimum accepted quality value |
| <i>nlowQ</i> | threshold on lowQ nucleotides (>= NOT allowed) |

Returns

0 if not used, 1 if accepted as is

4.50.3.12 `int Qtrim_global (Fq_read * seq, int left, int right, char type)`

trims left from the left and right from the right

Parameters

| | |
|--------------|--|
| <i>seq</i> | fastq read |
| <i>left</i> | number of nucleotides to be trimmed from the left |
| <i>right</i> | number of nucleotides to be trimmed from the right |
| <i>type</i> | char indicating the type of trimming (Q,A). |

Returns

2, since they are all accepted and trim

4.50.3.13 int trim_adapter (Fq_read * seq, Ad_seq * adap_list)

trims sequence based on presence of N nucleotides

if (adapter length < 16) -> search for seeds 8 nucleotides long else -> search for seeds 16 nucleotides long if (seed found) -> calculate score if score > threshold -> aligner found, trim / discard and exit. else -> search for seeds 8 nucleotides long

Parameters

| | |
|------------------|---------------------------|
| <i>seq</i> | pointer to Fq_read |
| <i>adap_list</i> | array of Ad_seq |

Returns

-1 error, 0 discarded, 1 accepted as is, 2 accepted and trimmed

Note

Global input parameters from par_TF are also used

4.50.3.14 int trim_sequenceN (Fq_read * seq)

trims sequence based on presence of N nucleotides

Parameters

| | |
|------------|------------|
| <i>seq</i> | fastq read |
|------------|------------|

Returns

-1 error, 0 discarded, 1 accepted as is, 2 accepted and trimmed

This function calls a different function depending on the method passed as input par_TF.trimN:

- **NO(0)**: accepts it as is, (1),
- **ALL(1)**: accepts it as is if NO N's found (1), rejects it otherwise (0),
- **ENDS(2)**: trims the ends and accepts it if it is longer than minL (2 if trimming, 1 if no trimming), rejects it otherwise (0),
- **STRIP(3)**: finds the longest N-free subsequence and trims it if it is at least minL nucleotides long (2 if trimming, 1 if no N's are found), rejects it otherwise (0).

4.50.3.15 int trim_sequenceQ (Fq_read * seq)

trims sequence based on lowQ base callings

Parameters

| | |
|------------|------------|
| <i>seq</i> | fastq read |
|------------|------------|

Returns

-1 error, 0 discarded, 1 accepted as is, 2 accepted and trimmed

This function calls a different function depending on the method passed as input par_TF.trimQ:

- **NO(0)**: accepts is as is , (1),
- **FRAC(1)**: accepts it if less than par_TF.nlowQ are found (1), rejects it otherwise (0),
- **ENDS(2)**: trims the ends and accepts it if it is longer than minL (2 if trimming, 1 if no trimming), rejects it otherwise (0),
- **ENDSFRAC(3)**: trims the ends and accepts if the remaining sequence is at least minL bases long and if it contains less than nlowQ lowQ nucleotides (2 if trimming, 1 if no trimming). Otherwise, it is rejected, (0).
- **GLOBAL(4)**: it trims globally globleft nucleotides from the left and globright from the right, (returns 2).

4.50.4 Variable Documentation**4.50.4.1 int Nencode**

global variable. Encoding for N's(\004, or \005)

4.50.4.2 lparam_trimFilter par_TF

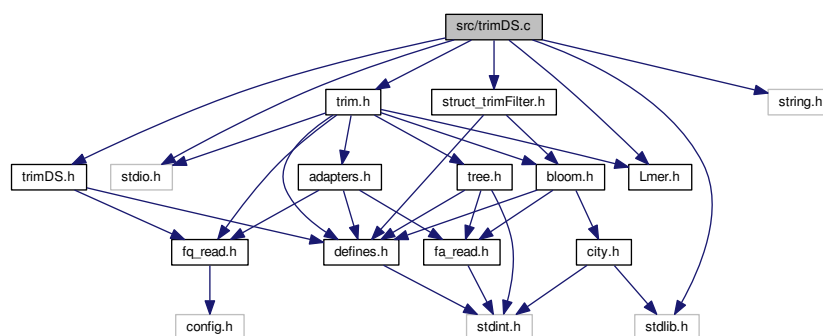
global variable: Input parameters trimFilter.

global variable: Input parameters of makeTree.

4.51 src/trimDS.c File Reference

trim adapters from double stranded data

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "trimDS.h"
#include "Lmer.h"
#include "trim.h"
#include "struct_trimFilter.h"
Include dependency graph for trimDS.c:
```



Functions

- `DS_adap init_DSadap` (char *ad1, char *ad2, int L1, int L2)
initialization of a DS_adap structure
- static void `pack_reads` (DS_adap *ptr_DSad, Fq_read *r1, Fq_read *r2)
pack reads with process_seq so that we can compute the edit distance between two subsequences. Read 1 will be packed once in the forward direction and Read 2 twice in the reverse direction with a shift of half a byte.
- static int `QtrimDS` (Fq_read *r1, Fq_read *r2, int L)
trims both reads reducing them to a length L.
- static double `obtain_scoreDS` (Fq_read *r1, int pos1, Fq_read *r2, int pos2)
obtains the score when comparing two subsequences of extended r1 and extended r2, starting in pos1 and pos2 respectively. The score is computed by adding log₁₀(4) when a match is observed and subtracting Q/10.0 when a mismatch is observed, with Q being the quality value. If there is a mismatch in a region where both read 1 and read 2 have qualities associated to the nucleotide under consideration, then the maximum of the quality values is subtracted.
- static int `alignDS_uint64` (Fq_read *r1, Fq_read *r2)
try to find adapter remnants in the reads r1 and r2 by aligning their extended versions (adapter1 + r1) vs rev_↔ comp(adapter2 + r2).
- int `trim_adapterDS` (DS_adap *ptr_DSad, Fq_read *r1, Fq_read *r2)
trim the sequences, discard them or keep them unchanged depending on them having adapters remnants.

Variables

- uint8_t `fw_1B` [256]
- uint8_t `bw_1B` [256]
- `lparam_trimFilter` par_TF

4.51.1 Detailed Description

trim adapters from double stranded data

Author

Paula Perez paulaperezrubio@gmail.com

Date

05.10.2017

4.51.2 Function Documentation

4.51.2.1 static int alignDS_uint64 (Fq_read * r1, Fq_read * r2) [static]

try to find adapter remnants in the reads r1 and r2 by aligning their extended versions (adapter1 + r1) vs rev_↔ comp(adapter2 + r2).

Parameters

| | |
|-----------|-------------------------------|
| <i>r1</i> | pointer to Fq_read for read 1 |
| <i>r2</i> | pointer to Fq_read for read 2 |

Returns

0 if the read is to be discarded, 1 if left as is, 2 if trimmed. Reads are trimmed if 2 is returned.

4.51.2.2 DS_adap init_DSadap (char * *ad1*, char * *ad2*, int *L1*, int *L2*)

initialization of a DS_adap structure

Parameters

| | |
|------------|---------------------------|
| <i>ad1</i> | adapter 1 sequence |
| <i>ad2</i> | adapter 2 sequence |
| <i>L1</i> | adapter 1 sequence length |
| <i>L2</i> | adapter 2 sequence length |

Returns

initialized DS_adap structure

4.51.2.3 static double obtain_scoreDS (Fq_read * r1, int pos1, Fq_read * r2, int pos2) [static]

obtains the score when comparing two subsequences of extended r1 and extended r2, starting in pos1 and pos2 respectively. The score is computed by adding log₁₀(4) when a match is observed and subtracting Q/10.0 when a mismatch is observed, with Q being the quality value. If there is a mismatch in a region where both read 1 and read 2 have qualities associated to the nucleotide under consideration, then the maximum of the quality values is subtracted.

Parameters

| | |
|-------------|---|
| <i>r1</i> | pointer to Fq_read for read 1 |
| <i>pos1</i> | position to start comparing in read 1, starting from 5' end of the extended sequence (adapter 1 + read 1) |
| <i>r2</i> | pointer to Fq_read for read 2 |
| <i>pos2</i> | position to start comparing in read 2, starting from 3' end of the extended sequence (adapter 2 + read 2) |

Returns

score associated to the comparison of the two strings

4.51.2.4 static void pack_reads (DS_adap * ptr_DSad, Fq_read * r1, Fq_read * r2) [static]

pack reads with process_seq so that we can compute the edit distance between two subsequences. Read 1 will be packed once in the forward direction and Read 2 twice in the reverse direction with a shift of half a byte.

Parameters

| | |
|-----------------|--|
| <i>ptr_DSad</i> | pointer to DS_adap structure, contains adapters sequences |
| <i>r1</i> | pointer to Fq_read for read 1, packed sequence stored here. |
| <i>r2</i> | pointer to Fq_read for read 2, packed sequences stored here. |

4.51.2.5 static int QtrimDS (Fq_read * r1, Fq_read * r2, int L) [static]

trims both reads reducing them to a length L.

Parameters

| | |
|-----------|-------------------------------|
| <i>r1</i> | pointer to Fq_read for read 1 |
| <i>r2</i> | pointer to Fq_read for read 2 |
| <i>L</i> | |

Returns

2 (we are trimming the sequences)

4.51.2.6 `int trim_adapterDS (DS_adap * ptr_DSad, Fq_read * r1, Fq_read * r2)`

trim the sequences, discard them or keep them unchanged depending on them having adapters remnants.

Returns

0 if the read is to be discarded, 1 if left as is, 2 if trimmed. Reads are trimmed if 2 is returned.

4.51.3 Variable Documentation

4.51.3.1 `uint8_t bw_1B[256]`

global variable. Lookup table.

4.51.3.2 `uint8_t fw_1B[256]`

global variable. Lookup table.

4.51.3.3 `lparam_trimFilter par_TF`

global variable. Input parameters.

global variable: Input parameters trimFilter.

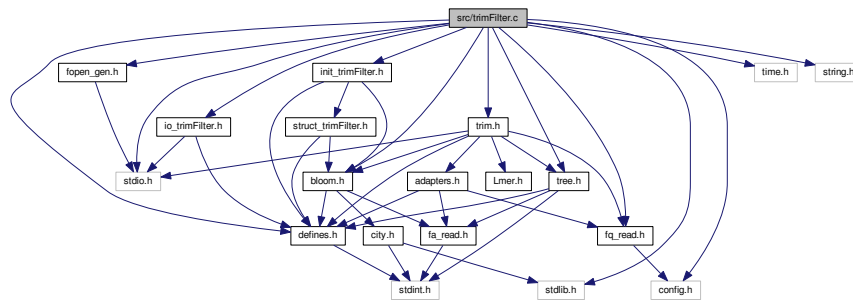
global variable: Input parameters of makeTree.

4.52 src/trimFilter.c File Reference

trimFilter main function

```
#include <stdio.h>
#include <time.h>
#include <string.h>
#include <stdlib.h>
#include "fq_read.h"
#include "fopen_gen.h"
#include "config.h"
#include "defines.h"
#include "init_trimFilter.h"
#include "io_trimFilter.h"
#include "tree.h"
#include "bloom.h"
#include "trim.h"
```

Include dependency graph for trimFilter.c:



Functions

- int [main](#) (int argc, char *argv[])
trimFilter main function

Variables

- uint64_t [alloc_mem](#) = 0
- [lparam_trimFilter](#) par_TF

4.52.1 Detailed Description

trimFilter main function

Author

Paula Perez paulaperezrubio@gmail.com

Date

25.08.2017 This file contains the trimFilter main function. See README_trimFilter.md for more details.

4.52.2 Variable Documentation

4.52.2.1 uint64_t alloc_mem = 0

global variable. Memory allocated in the heap.

4.52.2.2 lparam_trimFilter par_TF

global variable: Input parameters trimFilter.

4.53 src/trimFilterDS.c File Reference

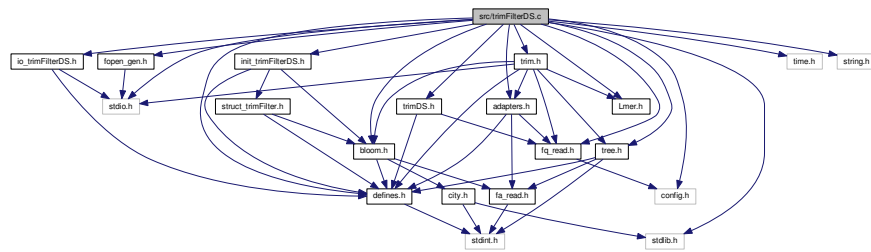
trimFilterDS main function

```

#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#include <string.h>
#include "defines.h"
#include "config.h"
#include "fopen_gen.h"
#include "trimDS.h"
#include "trim.h"
#include "tree.h"
#include "bloom.h"
#include "Lmer.h"
#include "adapters.h"
#include "fq_read.h"
#include "io_trimFilterDS.h"
#include "init_trimFilterDS.h"

```

Include dependency graph for trimFilterDS.c:



Functions

- int [main](#) (int argc, char *argv[])

contains trimfilterDS main function. See [README_trimFilterDS.md](#) for more details.

Variables

- uint64_t [alloc_mem](#) = 0
- [lparam_trimFilter](#) par_TF

4.53.1 Detailed Description

trimFilterDS main function

Author

Paula Perez paulaperezrubio@gmail.com

Date

25.08.2017 This file contains the trimFilterDS main function. See [README_trimFilterDS.md](#) and [README_trimFilter](#) for more details.

4.53.2 Variable Documentation

4.53.2.1 `uint64_t alloc_mem = 0`

global variable. Memory allocated in the heap.

4.53.2.2 `lparam_trimFilter par_TF`

global variable: Input parameters of makeTree.

Index

- nreads
 - statsinfo, [28](#)
- ntiles
 - statsinfo, [28](#)
- statsinfo, [27](#)
 - nreads, [28](#)
 - ntiles, [28](#)