Tabu Search TEA Reference Manual

Tabu search implementation to break TEA1 cipher algorithm

Generated by Doxygen 1.4.7

Sat Jul 1 18:40:18 2006

Contents

Chapter 1

Tabu Search TEA Data Structure Index

1.1 Tabu Search TEA Data Structures

Here are the data structures with brief descriptions:

Chapter 2

Tabu Search TEA File Index

2.1 Tabu Search TEA File List

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

Chapter 3

Tabu Search TEA Data Structure Documentation

3.1 best_result Struct Reference

Tabu Search best result implementation Best result for tabu search.

```
#include <types.h>
```

Data Fields

- unsigned long **key [2]**
- **float value**
- **unsigned int bit**

3.1.1 Detailed Description

Tabu Search best result implementation Best result for tabu search.

Definition at line 172 of file types.h.

3.1.2 Field Documentation

3.1.2.1 unsigned int best_result::bit

bit modified for this result

Definition at line 175 of file types.h.

Referenced by tabusearch().

3.1.2.2 unsigned long best_result::key[2]

key value for this result

Definition at line 173 of file types.h.

Referenced by print_end(), print_end_test_matrix(), print_iteration(), and tabusearch().

3.1.2.3 float best_result::value

evalutation value for this result

Definition at line 174 of file types.h.

Referenced by print_end(), print_end_test_matrix(), print_iteration(), and tabusearch().

3.2 container_node Struct Reference

input data structure Data structure that contains the input data (plain, cipher)

```
#include <types.h>
```

Data Fields

- unsigned long **plain_message [2]**
- **unsigned long cipher_message [2]**
- **int size_array**

3.2.1 Detailed Description

input data structure Data structure that contains the input data (plain, cipher)

Definition at line 39 of file types.h.

3.2.2 Field Documentation

3.2.2.1 unsigned long container_node::cipher_message[2]

64 bits cipher message splited in two 32 bits blocks

Definition at line 41 of file types.h.

Referenced by read_input().

3.2.2.2 unsigned long container_node::plain_message[2]

64 bits message splited in two 32 bits blocks

Definition at line 40 of file types.h.

Referenced by read_input().

3.2.2.3 int container_node::size_array

number of container nodes

Definition at line 42 of file types.h.

Referenced by evaluate(), main(), and read_input().

3.3 final_report Struct Reference

Final report for tabu search Structure that support data for final report generation.

```
#include <types.h>
```

Data Fields

- **best_result ∗ left**
- **best_result ∗ right**
- **time_t init_left**
- **time_t init_right**
- **time_t init_global**
- **time_t end_left**
- **time_t end_right**
- **time_t end_global**
- **clock_t clock_left**
- **clock_t clock_right**
- **clock_t clock_global**
- **unsigned int change_left_count**
- **unsigned int change_right_count**
- **unsigned int restart_left_control**
- **unsigned int restart_right_control**
- **unsigned int left_iter**
- **unsigned int right_iter**

3.3.1 Detailed Description

Final report for tabu search Structure that support data for final report generation.

Definition at line 208 of file types.h.

3.3.2 Field Documentation

3.3.2.1 unsigned int final_report::change_left_count

movement change for left block

Definition at line 220 of file types.h.

Referenced by print_end().

3.3.2.2 unsigned int final_report::change_right_count

movement change for right block

Definition at line 221 of file types.h.

Referenced by print_end().

3.3.2.3 clock_t final_report::clock_global

global program clock

Definition at line 219 of file types.h.

Referenced by print_end().

3.3.2.4 clock_t final_report::clock_left

program clock for left block

Definition at line 217 of file types.h.

Referenced by print_end(), and print_end_test_matrix().

3.3.2.5 clock_t final_report::clock_right

program clock for right block

Definition at line 218 of file types.h.

Referenced by print_end(), and print_end_test_matrix().

3.3.2.6 time_t final_report::end_global

end global time

Definition at line 216 of file types.h.

Referenced by print_end().

3.3.2.7 time_t final_report::end_left

end time for left block

Definition at line 214 of file types.h.

Referenced by print_end(), and print_end_test_matrix().

3.3.2.8 time_t final_report::end_right

end time for right block

Definition at line 215 of file types.h.

Referenced by print_end(), and print_end_test_matrix().

3.3.2.9 time_t final_report::init_global

initial global time

Definition at line 213 of file types.h.

Referenced by print_end().

3.3.2.10 time_t final_report::init_left

initial time for left block

Definition at line 211 of file types.h.

Referenced by print_end(), and print_end_test_matrix().

3.3.2.11 time_t final_report::init_right

initial time for right block

Definition at line 212 of file types.h.

Referenced by print_end(), and print_end_test_matrix().

3.3.2.12 best_result∗ final_report::left

best result for left block

Definition at line 209 of file types.h.

Referenced by print_end(), and print_end_test_matrix().

3.3.2.13 unsigned int final_report::left_iter

iterations for left block

Definition at line 224 of file types.h.

Referenced by print_end(), and print_end_test_matrix().

3.3.2.14 unsigned int final_report::restart_left_control

restarts for left block

Definition at line 222 of file types.h.

Referenced by print_end().

3.3.2.15 unsigned int final_report::restart_right_control

restarts for right block

Definition at line 223 of file types.h.

Referenced by print_end().

3.3.2.16 best_result∗ final_report::right

best result for left block

Definition at line 210 of file types.h.

Referenced by print_end(), and print_end_test_matrix().

3.3.2.17 unsigned int final_report::right_iter

iterations for right block

Definition at line 225 of file types.h.

Referenced by print_end(), and print_end_test_matrix().

3.4 full_test_input_opt Struct Reference

Structure that support input options for full test tabu search input options for full test tabu search.

`#include <types.h>`

Data Fields

- char **inputfile [255]**
- **char outfile [255]**
- **int save_output**
- **int have_input**
- **unsigned long long max_tabu_list_length**
- **unsigned long long max_tabu_iterations**
- **unsigned long long max_tabu_max_decrease**
- **unsigned long long max_change_move_limit**
- **unsigned long long init_tabu_list_length**
- **unsigned long long init_tabu_iterations**
- **unsigned long long init_tabu_max_decrease**
- **unsigned long long init_change_move_limit**
- **unsigned long long var_tabu_list_length**
- **unsigned long long var_tabu_iterations**
- **unsigned long long var_tabu_max_decrease**
- **unsigned long long var_change_move_limit**
- **float min_key_eval_percent**
- **float var_key_eval_percent**

3.4.1 Detailed Description

Structure that support input options for full test tabu search input options for full test tabu search.

Definition at line 102 of file types.h.

3.4.2 Field Documentation

3.4.2.1 int full_test_input_opt::have_input

if save output to a file

Definition at line 106 of file types.h.

Referenced by convert_full_test_opt_to_gen(), main(), and parse_full_test_arguments().

3.4.2.2 unsigned long long full_test_input_opt::init_change_move_limit

initial number of mistakes to change movement

Definition at line 114 of file types.h.

Referenced by convert_full_test_opt_to_gen(), main(), and parse_full_test_arguments().

3.4.2.3 unsigned long long full_test_input_opt::init_tabu_iterations

initial tabu iterations

Definition at line 112 of file types.h.

Referenced by convert_full_test_opt_to_gen(), main(), and parse_full_test_arguments().

3.4.2.4 unsigned long long full_test_input_opt::init_tabu_list_length

initial tabu list lenght

Definition at line 111 of file types.h.

Referenced by convert_full_test_opt_to_gen(), main(), and parse_full_test_arguments().

3.4.2.5 unsigned long long full_test_input_opt::init_tabu_max_decrease

initial performance max decrease

Definition at line 113 of file types.h.

Referenced by convert_full_test_opt_to_gen(), main(), and parse_full_test_arguments().

3.4.2.6 char full_test_input_opt::inputfile[255]

name for the input file

Definition at line 103 of file types.h.

Referenced by main(), and parse_full_test_arguments().

3.4.2.7 unsigned long long full_test_input_opt::max_change_move_limit

maximal number of mistakes to change movement

Definition at line 110 of file types.h.

Referenced by main(), and parse_full_test_arguments().

3.4.2.8 unsigned long long full_test_input_opt::max_tabu_iterations

maximal tabu iterations

Definition at line 108 of file types.h.

Referenced by main(), and parse_full_test_arguments().

3.4.2.9 unsigned long long full_test_input_opt::max_tabu_list_length

maximal tabu list lenght

Definition at line 107 of file types.h.

Referenced by main(), and parse_full_test_arguments().

3.4.2.10 unsigned long long full_test_input_opt::max_tabu_max_decrease

maximal tabu performance max decrease

Definition at line 109 of file types.h.

Referenced by main(), and parse_full_test_arguments().

3.4.2.11 float full_test_input_opt::min_key_eval_percent

initial percent for evaluation key

Definition at line 119 of file types.h.

Referenced by convert_full_test_opt_to_gen(), main(), and parse_full_test_arguments().

3.4.2.12 char full_test_input_opt::outfile[255]

name for the output file

Definition at line 104 of file types.h.

Referenced by convert_full_test_opt_to_gen(), and parse_full_test_arguments().

3.4.2.13 int full_test_input_opt::save_output

if save output to a file

Definition at line 105 of file types.h.

Referenced by convert_full_test_opt_to_gen(), and parse_full_test_arguments().

3.4.2.14 unsigned long long full_test_input_opt::var_change_move_limit

variation of mistakes number to change movement

Definition at line 118 of file types.h.

Referenced by main(), and parse_full_test_arguments().

3.4.2.15 float full_test_input_opt::var_key_eval_percent

variation of percent for evaluation key

Definition at line 120 of file types.h.

Referenced by main(), and parse_full_test_arguments().

3.4.2.16 unsigned long long full_test_input_opt::var_tabu_iterations

variation of tabu iterations

Definition at line 116 of file types.h.

Referenced by main(), and parse_full_test_arguments().

3.4.2.17 unsigned long long full_test_input_opt::var_tabu_list_length

variation of tabu list lenght

Definition at line 115 of file types.h.

Referenced by main(), and parse_full_test_arguments().

3.4.2.18 unsigned long long full_test_input_opt::var_tabu_max_decrease

variation of tabu performance max decrease

Definition at line 117 of file types.h.

Referenced by main(), and parse_full_test_arguments().

3.5 generate_options Struct Reference

generate plain-cipher input options Container for options in generation of random plain-cipher

```
#include <types.h>
```

Data Fields

- char **outfile [255]**
- **int userandom**
- **int usefile**
- **int usekey**
- **unsigned long seed**
- **unsigned long amount**
- **unsigned long key [4]**

3.5.1 Detailed Description

generate plain-cipher input options Container for options in generation of random plain-cipher

Definition at line 133 of file types.h.

3.5.2 Field Documentation

3.5.2.1 unsigned long generate_options::amount

amount of plain-cipher messages

Definition at line 139 of file types.h.

Referenced by main(), and parse_generate_arguments().

3.5.2.2 unsigned long generate_options::key[4]

if no randomd key, the key needs to be given

Definition at line 140 of file types.h.

Referenced by main(), and parse_generate_arguments().

3.5.2.3 char generate_options::outfile[255]

output filename

Definition at line 134 of file types.h.

Referenced by main(), and parse_generate_arguments().

3.5.2.4 unsigned long generate_options::seed

seed for random

Definition at line 138 of file types.h.

Referenced by main(), and parse_generate_arguments().

3.5.2.5 int generate_options::usefile

if outfile would be used, if not use STDOUT

Definition at line 136 of file types.h.

Referenced by main(), and parse_generate_arguments().

3.5.2.6 int generate_options::usekey

if key would be given

Definition at line 137 of file types.h.

Referenced by main(), and parse_generate_arguments().

3.5.2.7 int generate_options::userandom

if the key is random generated

Definition at line 135 of file types.h.

Referenced by parse_generate_arguments().

3.6 input_opt Struct Reference

Tabu search input options General Tabu search input options data structure.

```
#include <types.h>
```

Data Fields

- int **have_input**
- **int generate_report**
- **int save_output**
- **int paranoid_leve**
- **int middle_op**
- **int print_iter**
- **char inputfile [255]**
- **char outfile [255]**
- **unsigned long long tabu_list_length**
- **unsigned long long tabu_iterations**
- **unsigned long long tabu_max_decrease**
- **unsigned long long change_move_limit**
- **float key_eval_percent**

3.6.1 Detailed Description

Tabu search input options General Tabu search input options data structure.

Definition at line 55 of file types.h.

3.6.2 Field Documentation

3.6.2.1 unsigned long long input_opt::change_move_limit

tabu number of mistakes to change movement

Definition at line 67 of file types.h.

Referenced by convert_full_test_opt_to_gen(), create_params(), main(), and parse_ts_arguments().

3.6.2.2 int input_opt::generate_report

if generate report (ALWAYS TRUE)

Definition at line 57 of file types.h.

Referenced by convert_full_test_opt_to_gen(), and parse_ts_arguments().

3.6.2.3 int input_opt::have_input

if have input file

Definition at line 56 of file types.h.

Referenced by convert_full_test_opt_to_gen(), main(), and parse_ts_arguments().

3.6.2.4 char input_opt::inputfile[255]

name for the input file

Definition at line 62 of file types.h.

Referenced by main(), parse_ts_arguments(), and print_init().

3.6.2.5 float input_opt::key_eval_percent

percer for evaluation key

Definition at line 68 of file types.h.

Referenced by convert_full_test_opt_to_gen(), create_params(), main(), and parse_ts_arguments().

3.6.2.6 int input_opt::middle_op

if it's necessary to print middle operations

Definition at line 60 of file types.h.

Referenced by convert_full_test_opt_to_gen(), open_report(), and parse_ts_arguments().

3.6.2.7 char input_opt::outfile[255]

name for the output file

Definition at line 63 of file types.h.

Referenced by convert_full_test_opt_to_gen(), open_report(), parse_ts_arguments(), and print_init().

3.6.2.8 int input_opt::paranoid_leve

wich paranoid level would be used (DEFAULT 0)

Definition at line 59 of file types.h.

Referenced by convert_full_test_opt_to_gen(), evaluate(), move_alpha(), move_beta(), open_report(), parse_ts_arguments(), and tabusearch().

3.6.2.9 int input_opt::print_iter

if it's necessary to print iteration operations

Definition at line 61 of file types.h.

Referenced by convert_full_test_opt_to_gen(), open_report(), and parse_ts_arguments().

3.6.2.10 int input_opt::save_output

if save output to a file

Definition at line 58 of file types.h.

Referenced by close_report(), convert_full_test_opt_to_gen(), main(), open_report(), parse_ts_-arguments(), print_init(), and tabusearch().

3.6.2.11 unsigned long long input_opt::tabu_iterations

tabu iterations (DEFAULT 0)

Definition at line 65 of file types.h.

Referenced by convert_full_test_opt_to_gen(), create_params(), main(), and parse_ts_arguments().

3.6.2.12 unsigned long long input_opt::tabu_list_length

tabu list lenght (DEFAULT 0)

Definition at line 64 of file types.h.

Referenced by convert_full_test_opt_to_gen(), create_params(), main(), and parse_ts_arguments().

3.6.2.13 unsigned long long input_opt::tabu_max_decrease

tabu performance max decrease (DEFAULT 0)

Definition at line 66 of file types.h.

Referenced by convert_full_test_opt_to_gen(), create_params(), main(), and parse_ts_arguments().

3.7 output_report Struct Reference

Structure to support report and debug generation Support for report generation (initial al final report), and support for debugging options, function pointer aids change the function for each different context.

```
#include <types.h>
```

Data Fields

- **input_options ∗ options**
- **paranoid ∗ par**
- **paranoid ∗ pardeb**
- **FILE ∗ report_file**
- **void(∗ print_init)(ts_params ∗, input_options ∗, FILE ∗)**
- **void(∗ print_end)(final_report ∗, FILE ∗)**
- **void(∗ print_middle)(unsigned long, float, unsigned int, int, int, FILE ∗)**
- **void(∗ print_iteration)(best_result ∗, int, int, FILE ∗)**
- **void(∗ print_paranoid)(paranoid ∗, FILE ∗)**
- **void(∗ print_paranoid_move)(paranoid ∗, FILE ∗)**
- **void(∗ print_paranoid_eval)(paranoid ∗, FILE ∗)**
- **void(∗ print_paranoid_all)(paranoid ∗, FILE ∗)**

3.7.1 Detailed Description

Structure to support report and debug generation Support for report generation (initial al final report), and support for debugging options, function pointer aids change the function for each different context.

Definition at line 265 of file types.h.

3.7.2 Field Documentation

3.7.2.1 input_options∗ output_report::options

input options for this tabu serach program

Definition at line 266 of file types.h.

Referenced by close_report(), evaluate(), move_alpha(), move_beta(), open_report(), and tabusearch().

3.7.2.2 paranoid∗ output_report::par

Paranoid container for misc purpose

Definition at line 267 of file types.h.

Referenced by move_alpha(), move_beta(), and tabusearch().

3.7.2.3 paranoid∗ output_report::pardeb

Paranoid container for misc purpose

Definition at line 268 of file types.h.

Referenced by evaluate().

3.7.2.4 void(∗ output_report::print_end)(final_report ∗, FILE ∗)

print final report

Referenced by open_report(), and report_use_test_matrix().

3.7.2.5 void(∗ output_report::print_init)(ts_params ∗, input_options ∗, FILE ∗)

print the initial report

Referenced by open_report(), report_use_test_matrix(), and tabusearch().

3.7.2.6 void(∗ output_report::print_iteration)(best_result ∗, int, int, FILE ∗)

print tabu search iteration resutls

Referenced by open_report(), and tabusearch().

3.7.2.7 void(∗ output_report::print_middle)(unsigned long, float, unsigned int, int, int, FILE ∗)

print tabu search middle operations

Referenced by open_report().

3.7.2.8 void(∗ output_report::print_paranoid)(paranoid ∗, FILE ∗)

paranoid debug level 1, print tabu op

Referenced by open_report(), and tabusearch().

3.7.2.9 void(∗ output_report::print_paranoid_all)(paranoid ∗, FILE ∗)

paranoid debug level 4, print key kegeneration for key eval

Referenced by evaluate(), and open_report().

3.7.2.10 void(∗ output_report::print_paranoid_eval)(paranoid ∗, FILE ∗)

paranoid debug level 3, print evaluation function

Referenced by evaluate(), and open_report().

3.7.2.11 void(∗ output_report::print_paranoid_move)(paranoid ∗, FILE ∗)

paranoid debug level 2, print movement

Referenced by move_alpha(), move_beta(), and open_report().

3.7.2.12 FILE∗ output_report::report_file

where to write reports

Definition at line 269 of file types.h.

Referenced by close_report(), main(), open_report(), and tabusearch().

3.8 paranoid_level Struct Reference

Paranoid debug support Structure that support paranoid debug for tabu search operations, in order to work each attribute has a different meaning for each different context.

```
#include <types.h>
```

Data Fields

- unsigned long **key**
- **float score**
- **int block**
- **unsigned long evalkey**
- **tabu_list * list**
- **unsigned int tabu**
- **unsigned int bit**
- **unsigned int percent**
- **unsigned int zeros**
- **unsigned int ones**
- **unsigned int element**

3.8.1 Detailed Description

Paranoid debug support Structure that support paranoid debug for tabu search operations, in order to work each attribute has a different meaning for each different context.

Definition at line 239 of file types.h.

3.9 tabu_list Struct Reference

Tabu List implemetation data structure The tabu list, its a circular linked list which contains forbidden movements.

```
#include <types.h>
```

Data Fields

- unsigned long **key**
- **unsigned int bit_position**
- **float value**
- **unsigned int name**
- **tabu_list ∗ next**

3.9.1 Detailed Description

Tabu List implemetation data structure The tabu list, its a circular linked list which contains forbidden movements.

Definition at line 153 of file types.h.

3.9.2 Field Documentation

3.9.2.1 unsigned int tabu_list::bit_position

bit modification for this tabu

Definition at line 155 of file types.h.

Referenced by create_tabu_list(), move_alpha(), move_beta(), print_paranoid(), and restart_tabu().

3.9.2.2 unsigned long tabu_list::key

key value for this tabu

Definition at line 154 of file types.h.

Referenced by create_tabu_list(), print_paranoid(), and restart_tabu().

3.9.2.3 unsigned int tabu_list::name

name for this element

Definition at line 157 of file types.h.

Referenced by create_tabu_list(), free_tabu(), print_paranoid(), and restart_tabu().

3.9.2.4 struct tabu_list∗ tabu_list::next

next element in the tabu list

Definition at line 158 of file types.h.

Referenced by create_tabu_list(), free_tabu(), move_alpha(), move_beta(), print_paranoid(), and restart_-tabu().

3.9.2.5 float tabu_list::value

evaluation value for this tabu

Definition at line 156 of file types.h.

Referenced by create_tabu_list(), print_paranoid(), and restart_tabu().

3.10 tabu_movement Struct Reference

Structure to support tabu search morphic movement Function pointer aids to support multiple tabu search movements, this structure provides the mechanism to link movement and tabu search call.

```
#include <types.h>
```

Data Fields

- void(∗ **move**)(**unsigned long** ∗, **tabu_list** ∗, **unsigned long** ∗, **unsigned int** ∗, **unsigned int** ∗, **best_result** ∗, **ts_params** ∗, **output_report** ∗, **cipher_cont** ∗, **int**)

3.10.1 Detailed Description

Structure to support tabu search morphic movement Function pointer aids to support multiple tabu search movements, this structure provides the mechanism to link movement and tabu search call.

Definition at line 291 of file types.h.

3.10.2 Field Documentation

3.10.2.1 void(∗ tabu_movement::move)(unsigned long ∗, tabu_list ∗, unsigned long ∗, unsigned int ∗, unsigned int ∗, best_result ∗, ts_params ∗, output_report ∗, cipher_cont ∗, int)

Tabu search movement

Referenced by tabusearch().

3.11 tabu_search_params Struct Reference

Tabu Search parameters A set of parameters to perform tabu search.

```
#include <types.h>
```

Data Fields

- unsigned long long **tabu_list_length**
- **unsigned long long tabu_iterations**
- **unsigned long long tabu_max_decrease**
- **unsigned long long change_move_limit**
- **float key_eval_percent**

3.11.1 Detailed Description

Tabu Search parameters A set of parameters to perform tabu search.

Definition at line 189 of file types.h.

3.11.2 Field Documentation

3.11.2.1 unsigned long long tabu_search_params::change_move_limit

mistakes for change the movement

Definition at line 193 of file types.h.

Referenced by create_params(), print_init(), and tabusearch().

3.11.2.2 float tabu_search_params::key_eval_percent

percent of equality of key bits

Definition at line 194 of file types.h.

Referenced by create_params(), evaluate(), print_init(), and print_init_test_matrix().

3.11.2.3 unsigned long long tabu_search_params::tabu_iterations

maximal number of iterations

Definition at line 191 of file types.h.

Referenced by create_params(), print_init(), print_init_test_matrix(), and tabusearch().

3.11.2.4 unsigned long long tabu_search_params::tabu_list_length

tabu list length

Definition at line 190 of file types.h.

Referenced by create_params(), create_tabu_list(), move_alpha(), move_beta(), print_init(), print_init_-test_matrix(), and restart_tabu().

3.11.2.5 unsigned long long tabu_search_params::tabu_max_decrease

performance decreace limit

Definition at line 192 of file types.h.

Referenced by create_params(), print_init(), and print_init_test_matrix().

3.12 test_input_opt Struct Reference

Cipher Test input options Cipher Test input options data.

```
#include <types.h>
```

Data Fields

- char **inputfile [255]**
- **char outfile [255]**
- **int save_output**
- **int have_input**
- **unsigned long long key1**
- **unsigned long long key2**
- **unsigned long long key3**
- **unsigned long long key4**

3.12.1 Detailed Description

Cipher Test input options Cipher Test input options data.

Definition at line 81 of file types.h.

3.12.2 Field Documentation

3.12.2.1 int test_input_opt::have_input

if have input file

Definition at line 85 of file types.h.

Referenced by main(), and parse_test_arguments().

3.12.2.2 char test_input_opt::inputfile[255]

name for the input file

Definition at line 82 of file types.h.

Referenced by main(), and parse_test_arguments().

3.12.2.3 unsigned long long test_input_opt::key2

< block 1 for input key

Definition at line 87 of file types.h.

Referenced by main(), and parse_test_arguments().

3.12.2.4 unsigned long long test_input_opt::key3

< block 2 for input key

Definition at line 88 of file types.h.

Referenced by main(), and parse_test_arguments().

3.12.2.5 unsigned long long test_input_opt::key4

< block 3 for input key

Definition at line 89 of file types.h.

Referenced by main(), and parse_test_arguments().

3.12.2.6 char test_input_opt::outfile[255]

name for the output file

Definition at line 83 of file types.h.

Referenced by parse_test_arguments().

3.12.2.7 int test_input_opt::save_output

if save output to a file

Definition at line 84 of file types.h.

Referenced by parse_test_arguments().

Chapter 4

Tabu Search TEA File Documentation

4.1 /Users/freddy/Documents/Research_proyects/TEA/code/generate.c File Reference

Generation of test set for TEA Tabu Search.

```
#include <stdio.h>
#include <stdlib.h>
#include "tea.h"
#include "io.h"
#include "types.h"
```

Functions

- int **main (int argc, char ∗∗argv)**

4.1.1 Detailed Description

Generation of test set for TEA Tabu Search.

Author:

Freddy Mun∼oz Ramirez <frmunoz(at)inf.utfsm.cl>

Date:

Autumn 2006
This code can be re-distributed under MIT License

Definition in file **generate.c.**

4.1.2 Function Documentation

4.1.2.1 int main (int *argc*, char $**$ *argv*)

Generate a test set with TEA cipher

Parameters:

arcg	number of input arguments
argv	input argument string

Definition at line 19 of file generate.c.

References generate_options::amount, generate_options::key, MALLOC, generate_options::outfile, parse_generate_arguments(), print_generate_options(), generate_options::seed, tea_encrypt(), generate_options::usefile, and generate_options::usekey.

4.2 /Users/freddy/Documents/Research_proyects/TEA/code/io.c File Reference

input output functions

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "io.h"
#include "types.h"
#include "tabu_search.h"
```

Functions

- **cipher_cont ∗ read_input (char ∗filename)**
- **input_options ∗ convert_full_test_opt_to_gen (full_test_input_options ∗options)**
- **void parse_full_test_arguments (int argc, char ∗∗argv, full_test_input_options ∗options)**
- **void parse_test_arguments (int argc, char ∗∗argv, test_input_options ∗options)**
- **void parse_generate_arguments (int argc, char ∗∗argv, generate_options ∗options)**
- **void parse_ts_arguments (int argc, char ∗∗argv, input_options ∗options)**
- **void print_full_test_options (char ∗name)**
- **void print_ts_options (char ∗name)**
- **void print_test_options (char ∗name)**
- **void print_generate_options (char ∗name)**
- **output_report ∗ open_report (input_options ∗options)**
- **void close_report (output_report ∗report)**
- **void report_use_test_matrix (output_report ∗report)**
- **void print_init (ts_params ∗params, input_options ∗options, FILE ∗file)**
- **void print_init_test_matrix (ts_params ∗params, input_options ∗options, FILE ∗file)**
- **void print_end_test_matrix (final_report ∗report, FILE ∗file)**
- **void print_mold_test_matrix (FILE ∗file)**
- **void print_end (final_report ∗report, FILE ∗file)**
- **void print_middle (unsigned long key, float score, unsigned int bit, int block, int tabuname, FILE ∗file)**
- **void print_iteration (best_result ∗best, int iter, int block, FILE ∗file)**
- **void print_paranoid (paranoid ∗paranoid, FILE ∗file)**
- **void print_paranoid_move (paranoid ∗paranoid, FILE ∗file)**
- **void print_paranoid_eval (paranoid ∗paranoid, FILE ∗file)**
- **void print_paranoid_all (paranoid ∗paranoid, FILE ∗file)**
- **void void_print_middle (unsigned long key, float score, unsigned int bit, int block, int tabuname, FILE ∗fp)**
- **void void_print_iteration (best_result ∗best, int block, int iter, FILE ∗fp)**
- **void void_print_paranoid (paranoid ∗paranoid, FILE ∗fp)**
- **void void_print_paranoid_move (paranoid ∗paranoid, FILE ∗fp)**
- **void void_print_paranoid_eval (paranoid ∗paranoid, FILE ∗fp)**
- **void void_print_paranoid_all (paranoid ∗paranoid, FILE ∗fp)**

4.2.1 Detailed Description

input output functions

Author:

Freddy Mun∼oz Ramirez <frmunoz(at)inf.utfsm.cl>

Date:

Autumn 2006
This code can be re-distributed under MIT License

Definition in file io.c.

4.2.2 Function Documentation

4.2.2.1 void close_report (output_report ∗ *report*)

Close output file for given report

Parameters:

report Report generator

Definition at line 662 of file io.c.

References output_report::options, output_report::report_file, and input_opt::save_output.

Referenced by main().

4.2.2.2 input_options∗ convert_full_test_opt_to_gen (full_test_input_options ∗ *options*)

Convert full_test_input_options to input_options

Parameters:

options Tabu Serach general input options

Definition at line 80 of file io.c.

References input_opt::change_move_limit, FALSE, input_opt::generate_report, input_opt::have_-input, full_test_input_opt::have_input, full_test_input_opt::init_change_move_limit, full_test_-input_opt::init_tabu_iterations, full_test_input_opt::init_tabu_list_length, full_test_input_-opt::init_tabu_max_decrease, input_opt::key_eval_percent, MALLOC, input_opt::middle_op, full_test_input_opt::min_key_eval_percent, input_opt::outfile, full_test_input_opt::outfile, input_-opt::paranoid_leve, input_opt::print_iter, input_opt::save_output, full_test_input_opt::save_-output, input_opt::tabu_iterations, input_opt::tabu_list_length, input_opt::tabu_max_decrease, and TRUE.

Referenced by main().

4.2.2.3 output_report∗ open_report (input_options ∗ *options*)

Create a new report creator and bind the correct function in accord with options

Parameters:

options Tabu Search general input options

Definition at line 580 of file io.c.

References MALLOC, input_opt::middle_op, output_report::options, input_opt::outfile, input_-opt::paranoid_leve, output_report::print_end, print_end(), output_report::print_init, print_init(), input_opt::print_iter, output_report::print_iteration, print_iteration(), output_report::print_-middle, print_middle(), output_report::print_paranoid, print_paranoid(), output_report::print_-paranoid_all, print_paranoid_all(), output_report::print_paranoid_eval, print_paranoid_eval(), output_report::print_paranoid_move, print_paranoid_move(), output_report::report_file, input_-opt::save_output, void_print_iteration(), void_print_middle(), void_print_paranoid(), void_print_-paranoid_all(), void_print_paranoid_eval(), and void_print_paranoid_move().

Referenced by main().

4.2.2.4 void parse_full_test_arguments (int *argc*, char ** *argv*, full_test_input_options * *options*)

Parse full test input arguments and mold it into full_test_input_options structure

Parameters:

argc ANSI C program input parameters count
argv ANSI C program input string
options Where to put options in parameters

Definition at line 108 of file io.c.

References FALSE, full_test_input_opt::have_input, full_test_input_opt::init_change_move_limit, full_test_input_opt::init_tabu_iterations, full_test_input_opt::init_tabu_list_length, full_test_-input_opt::init_tabu_max_decrease, full_test_input_opt::inputfile, full_test_input_opt::max_-change_move_limit, full_test_input_opt::max_tabu_iterations, full_test_input_opt::max_tabu_list_-length, full_test_input_opt::max_tabu_max_decrease, full_test_input_opt::min_key_eval_percent, full_test_input_opt::outfile, full_test_input_opt::save_output, TRUE, full_test_input_opt::var_-change_move_limit, full_test_input_opt::var_key_eval_percent, full_test_input_opt::var_tabu_-iterations, full_test_input_opt::var_tabu_list_length, and full_test_input_opt::var_tabu_max_-decrease.

Referenced by main().

4.2.2.5 void parse_generate_arguments (int *argc*, char ** *argv*, generate_options * *options*)

Parse TEA generation input arguments and mold it into generate_iptions structure

Parameters:

argc ANSI C program input parameters count
argv ANSI C program input string
options Where to put options in parameters

Definition at line 330 of file io.c.

References generate_options::amount, FALSE, generate_options::key, generate_options::outfile, print_generate_options(), generate_options::seed, TRUE, generate_options::usefile, generate_-options::usekey, and generate_options::userandom.

Referenced by main().

4.2.2.6 void parse_test_arguments (int *argc*, char ∗∗ *argv*, test_input_options ∗ *options*)

Parse cipher test input arguments and mold it into test_input_options structure

Parameters:

argc ANSI C program input parameters count

argv ANSI C program input string

options Where to put options in parameters

Definition at line 278 of file io.c.

References FALSE, test_input_opt::have_input, test_input_opt::inputfile, test_input_opt::key1, test_input_opt::key2, test_input_opt::key3, test_input_opt::key4, test_input_opt::outfile, test_-input_opt::save_output, and TRUE.

Referenced by main().

4.2.2.7 void parse_ts_arguments (int *argc*, char ∗∗ *argv*, input_options ∗ *options*)

Parse Tabu Search input arguments and mold it into input_options structure

Parameters:

argc ANSI C program input parameters count

argv ANSI C program input string

options Where to put options in parameters

Definition at line 392 of file io.c.

References input_opt::change_move_limit, FALSE, input_opt::generate_report, input_opt::have_-input, input_opt::inputfile, input_opt::key_eval_percent, input_opt::middle_op, input_opt::outfile, input_opt::paranoid_leve, input_opt::print_iter, input_opt::save_output, input_opt::tabu_-iterations, input_opt::tabu_list_length, input_opt::tabu_max_decrease, and TRUE.

Referenced by main().

4.2.2.8 void print_end (final_report ∗ *report*, FILE ∗ *file*)

Print final report of Tabu Search

Parameters:

report Report generator

file File to print

Definition at line 741 of file io.c.

References final_report::change_left_count, final_report::change_right_count, final_report::clock_-global, final_report::clock_left, final_report::clock_right, final_report::end_global, final_-report::end_left, final_report::end_right, final_report::init_global, final_report::init_left, final_-report::init_right, best_result::key, final_report::left, final_report::left_iter, final_report::restart_-left_control, final_report::restart_right_control, final_report::right, final_report::right_iter, and best_result::value.

Referenced by open_report().

4.2.2.9 void print_end_test_matrix (final_report ∗ *report*, FILE ∗ *file*)

Print each step Tabu Search for test matrix

Parameters:

report Report generator
file File to print

Definition at line 718 of file io.c.

References final_report::clock_left, final_report::clock_right, final_report::end_left, final_-report::end_right, final_report::init_left, final_report::init_right, best_result::key, final_report::left, final_report::left_iter, final_report::right, final_report::right_iter, and best_result::value.

Referenced by report_use_test_matrix().

4.2.2.10 void print_full_test_options (char ∗ *name*)

Print full test usage and options

Parameters:

name Name for executable

Definition at line 500 of file io.c.

Referenced by main().

4.2.2.11 void print_generate_options (char ∗ *name*)

Print generate usage and options

Parameters:

name Name for executable

Definition at line 573 of file io.c.

Referenced by main(), and parse_generate_arguments().

4.2.2.12 void print_init (ts_params ∗ *params*, input_options ∗ *options*, FILE ∗ *file*)

Print initial asignation for Tabu Search

Parameters:

params Paramenter of Tabu Search
options Tabu Search general input options
file File to print

Definition at line 683 of file io.c.

References tabu_search_params::change_move_limit, FALSE, input_opt::inputfile, tabu_search_-params::key_eval_percent, input_opt::outfile, input_opt::save_output, tabu_search_params::tabu_-iterations, tabu_search_params::tabu_list_length, and tabu_search_params::tabu_max_decrease.

Referenced by open_report().

4.2.2.13 void print_init_test_matrix (ts_params ∗ *params*, input_options ∗ *options*, FILE ∗ *file*)

Print initial asignation for Tabu Search in output text matrix

Parameters:

params Paramenter of Tabu Search

options Tabu Search general input options

file File to print

Definition at line 709 of file io.c.

References tabu_search_params::key_eval_percent, tabu_search_params::tabu_iterations, tabu_-search_params::tabu_list_length, and tabu_search_params::tabu_max_decrease.

Referenced by report_use_test_matrix().

4.2.2.14 void print_iteration (best_result ∗ *best*, int *iter*, int *block*, FILE ∗ *file*)

Print iteration step values

Parameters:

best Best result until this iteration

iter Iteration number

block Block for this iteration {LEFT,RIGHT}

file File to print

Definition at line 781 of file io.c.

References best_result::key, and best_result::value.

Referenced by open_report().

4.2.2.15 void print_middle (unsigned long *key*, float *score*, unsigned int *bit*, int *block*, int *tabuname*, FILE ∗ *file*)

Print step by step middle operation for Tabu Search

Parameters:

key Key in given step

score Score for given step

bit Bit changed in this step

block Block for this step {LEFT,RIGHT}

Definition at line 770 of file io.c.

Referenced by open_report().

4.2.2.16 void print_mold_test_matrix (FILE ∗ *file*)

Print initial significant keyswords for test matrix

Parameters:

file File to print

Definition at line 732 of file io.c.

Referenced by main().

4.2.2.17 void print_paranoid (paranoid ∗ *paranoid*, FILE ∗ *file*)

Print really paranoid step debug, include key movement values, and tabu list

Parameters:

paranoid Paranoid data
file File to print

Definition at line 790 of file io.c.

References tabu_list::bit_position, paranoid_level::block, paranoid_level::key, tabu_list::key, paranoid_level::list, tabu_list::name, tabu_list::next, paranoid_level::tabu, and tabu_list::value.

Referenced by open_report().

4.2.2.18 void print_paranoid_all (paranoid ∗ *paranoid*, FILE ∗ *file*)

Print really paranoid step debug for evaluation function, include block key generation

Parameters:

paranoid Paranoid data
file File to print

Definition at line 826 of file io.c.

References paranoid_level::bit, paranoid_level::block, paranoid_level::evalkey, and paranoid_-level::key.

Referenced by open_report().

4.2.2.19 void print_paranoid_eval (paranoid ∗ *paranoid*, FILE ∗ *file*)

Print really paranoid step debug for evaluation function

Parameters:

paranoid Paranoid data
file File to print

Definition at line 817 of file io.c.

References paranoid_level::bit, paranoid_level::element, paranoid_level::ones, paranoid_-level::percent, and paranoid_level::zeros.

Referenced by open_report().

4.2.2.20 void print_paranoid_move (paranoid ∗ *paranoid*, FILE ∗ *file*)

Print really paranoid step debug, include movement middle key values and evaluation value

Parameters:

paranoid Paranoid data

file File to print

Definition at line 808 of file io.c.

References paranoid_level::bit, paranoid_level::block, paranoid_level::key, and paranoid_-level::score.

Referenced by open_report().

4.2.2.21 void print_test_options (char ∗ *name*)

Print cipher test usage and options

Parameters:

name Name for executable

Definition at line 559 of file io.c.

Referenced by main().

4.2.2.22 void print_ts_options (char ∗ *name*)

Print Tabu Search usage and options

Parameters:

name Name for executable

Definition at line 529 of file io.c.

Referenced by main().

4.2.2.23 cipher_cont∗ read_input (char ∗ *filename*)

Read input file and put the content into corresponding structure

Parameters:

filename Filename of input file

Definition at line 20 of file io.c.

References container_node::cipher_message, MAX_FILE_LINES, NMALLOC, container_-node::plain_message, and container_node::size_array.

Referenced by main().

4.2.2.24 void report_use_test_matrix (output_report ∗ *report*)

Change report binding to use test matrix

Parameters:

report Report generator

Definition at line 672 of file io.c.

References output_report::print_end, print_end_test_matrix(), output_report::print_init, and print_init_test_matrix().

Referenced by main().

4.2.2.25 void void_print_iteration (best_result ∗ *best*, int *block*, int *iter*, FILE ∗ *fp*)

DO NOTHING!

Parameters:

best Best result until this iteration

iter Iteration number

block Block for this iteration {LEFT,RIGHT}

file File to print

Definition at line 849 of file io.c.

Referenced by open_report().

4.2.2.26 void void_print_middle (unsigned long *key*, float *score*, unsigned int *bit*, int *block*, int *tabuname*, FILE ∗ *fp*)

DO NOTHING!

Parameters:

key Key in given step

score Score for given step

bit Bit changed in this step

block Block for this step {LEFT,RIGHT}

Definition at line 839 of file io.c.

Referenced by open_report().

4.2.2.27 void void_print_paranoid (paranoid ∗ *paranoid*, FILE ∗ *fp*)

DO NOTHING!

Parameters:

paranoid Paranoid data

file File to print

Definition at line 857 of file io.c.

Referenced by open_report().

4.2.2.28 void void_print_paranoid_all (paranoid ∗ *paranoid*, FILE ∗ *fp*)

DO NOTHING!

Parameters:

paranoid **Paranoid data**

file **File to print**

Definition at line 881 of file io.c.

Referenced by open_report().

4.2.2.29 void void_print_paranoid_eval (paranoid ∗ *paranoid*, FILE ∗ *fp*)

DO NOTHING!

Parameters:

paranoid **Paranoid data**

file **File to print**

Definition at line 873 of file io.c.

Referenced by open_report().

4.2.2.30 void void_print_paranoid_move (paranoid ∗ *paranoid*, FILE ∗ *fp*)

DO NOTHING!

Parameters:

paranoid **Paranoid data**

file **File to print**

Definition at line 865 of file io.c.

Referenced by open_report().

4.3 /Users/freddy/Documents/Research_proyects/TEA/code/io.h File Reference

input output constant and functions definition

```
#include "types.h"
```

Defines

- **#define MAX_FILE_LINES 6000**

Functions

- **cipher_cont ∗ read_input (char ∗)**
- **void parse_ts_arguments (int, char ∗∗, input_options ∗)**
- **void parse_test_arguments (int, char ∗∗, test_input_options ∗)**
- **void parse_full_test_arguments (int, char ∗∗, full_test_input_options ∗)**
- **void parse_generate_arguments (int, char ∗∗, generate_options ∗)**
- **void print_ts_options (char ∗)**
- **void print_test_options (char ∗)**
- **void print_full_test_options (char ∗)**
- **void print_generate_options (char ∗)**
- **void print_init (ts_params ∗, input_options ∗, FILE ∗)**
- **void print_end (final_report ∗, FILE ∗)**
- **void print_init_test_matrix (ts_params ∗, input_options ∗options, FILE ∗)**
- **void print_end_test_matrix (final_report ∗, FILE ∗)**
- **void print_middle (unsigned long, float, unsigned int, int, int, FILE ∗)**
- **void print_iteration (best_result ∗, int, int, FILE ∗)**
- **void print_paranoid (paranoid ∗, FILE ∗)**
- **void print_paranoid_move (paranoid ∗, FILE ∗)**
- **void print_paranoid_eval (paranoid ∗, FILE ∗)**
- **void print_paranoid_all (paranoid ∗, FILE ∗)**
- **void void_print_middle (unsigned long, float, unsigned int, int, int, FILE ∗)**
- **void void_print_iteration (best_result ∗, int, int, FILE ∗)**
- **void void_print_paranoid (paranoid ∗, FILE ∗)**
- **void void_print_paranoid_move (paranoid ∗, FILE ∗)**
- **void void_print_paranoid_eval (paranoid ∗, FILE ∗)**
- **void void_print_paranoid_all (paranoid ∗, FILE ∗)**
- **output_report ∗ open_report (input_options ∗options)**
- **void close_report (output_report ∗)**
- **void report_use_test_matrix (output_report ∗)**
- **void print_mold_test_matrix (FILE ∗)**
- **input_options ∗ convert_full_test_opt_to_gen (full_test_input_options ∗)**

4.3.1 Detailed Description

input output constant and functions definition

Author:

Freddy Mun∼oz Ramirez <frmunoz(at)inf.utfsm.cl>

Date:

Autumn 2006
This code can be re-distributed under MIT License

Definition in file io.h.

4.3.2 Define Documentation

4.3.2.1 #define MAX_FILE_LINES 6000

Maximal number of lines in input file

Definition at line 16 of file io.h.

Referenced by read_input().

4.3.3 Function Documentation

4.3.3.1 void close_report (output_report ∗ *report*)

Close output file for given report

Parameters:

report Report generator

Definition at line 662 of file io.c.

References output_report::options, output_report::report_file, and input_opt::save_output.

Referenced by main().

4.3.3.2 input_options∗ convert_full_test_opt_to_gen (full_test_input_options ∗ *options*)

Convert full_test_input_options to input_options

Parameters:

options Tabu Serach general input options

Definition at line 80 of file io.c.

References input_opt::change_move_limit, FALSE, input_opt::generate_report, full_test_input_-opt::have_input, input_opt::have_input, full_test_input_opt::init_change_move_limit, full_-test_input_opt::init_tabu_iterations, full_test_input_opt::init_tabu_list_length, full_test_input_-opt::init_tabu_max_decrease, input_opt::key_eval_percent, MALLOC, input_opt::middle_op, full_test_input_opt::min_key_eval_percent, full_test_input_opt::outfile, input_opt::outfile, input_-opt::paranoid_leve, input_opt::print_iter, full_test_input_opt::save_output, input_opt::save_-output, input_opt::tabu_iterations, input_opt::tabu_list_length, input_opt::tabu_max_decrease, and TRUE.

Referenced by main().

4.3.3.3 output_report∗ open_report (input_options ∗ *options*)

Create a new report creator and bind the correct function in accord with options

Parameters:

options Tabu Search general input options

Definition at line 580 of file io.c.

References MALLOC, input_opt::middle_op, output_report::options, input_opt::outfile, input_-opt::paranoid_leve, print_end(), output_report::print_end, print_init(), output_report::print_init, input_opt::print_iter, print_iteration(), output_report::print_iteration, print_middle(), output_-report::print_middle, print_paranoid(), output_report::print_paranoid, print_paranoid_all(), output_report::print_paranoid_all, print_paranoid_eval(), output_report::print_paranoid_-eval, print_paranoid_move(), output_report::print_paranoid_move, output_report::report_file, input_opt::save_output, void_print_iteration(), void_print_middle(), void_print_paranoid(), void_print_paranoid_all(), void_print_paranoid_eval(), and void_print_paranoid_move().

Referenced by main().

4.3.3.4 void parse_full_test_arguments (int *argc*, char ∗∗ *argv*, full_test_input_options ∗ *options*)

Parse full test input arguments and mold it into full_test_input_options structure

Parameters:

argc ANSI C program input parameters count

argv ANSI C program input string

options Where to put options in parameters

Definition at line 108 of file io.c.

References FALSE, full_test_input_opt::have_input, full_test_input_opt::init_change_move_limit, full_test_input_opt::init_tabu_iterations, full_test_input_opt::init_tabu_list_length, full_test_-input_opt::init_tabu_max_decrease, full_test_input_opt::inputfile, full_test_input_opt::max_-change_move_limit, full_test_input_opt::max_tabu_iterations, full_test_input_opt::max_tabu_list_-length, full_test_input_opt::max_tabu_max_decrease, full_test_input_opt::min_key_eval_percent, full_test_input_opt::outfile, full_test_input_opt::save_output, TRUE, full_test_input_opt::var_-change_move_limit, full_test_input_opt::var_key_eval_percent, full_test_input_opt::var_tabu_-iterations, full_test_input_opt::var_tabu_list_length, and full_test_input_opt::var_tabu_max_-decrease.

Referenced by main().

4.3.3.5 void parse_generate_arguments (int *argc*, char ∗∗ *argv*, generate_options ∗ *options*)

Parse TEA generation input arguments and mold it into generate_iptions structure

Parameters:

argc ANSI C program input parameters count

argv ANSI C program input string

options Where to put options in parameters

Definition at line 330 of file io.c.

References generate_options::amount, FALSE, generate_options::key, generate_options::outfile, print_generate_options(), generate_options::seed, TRUE, generate_options::usefile, generate_-options::usekey, and generate_options::userandom.

Referenced by main().

4.3.3.6 void parse_test_arguments (int *argc*, char ∗∗ *argv*, test_input_options ∗ *options*)

Parse cipher test input arguments and mold it into test_input_options structure

Parameters:

argc ANSI C program input parameters count

argv ANSI C program input string

options Where to put options in parameters

Definition at line 278 of file io.c.

References FALSE, test_input_opt::have_input, test_input_opt::inputfile, test_input_opt::key1, test_input_opt::key2, test_input_opt::key3, test_input_opt::key4, test_input_opt::outfile, test_-input_opt::save_output, and TRUE.

Referenced by main().

4.3.3.7 void parse_ts_arguments (int *argc*, char ∗∗ *argv*, input_options ∗ *options*)

Parse Tabu Search input arguments and mold it into input_options structure

Parameters:

argc ANSI C program input parameters count

argv ANSI C program input string

options Where to put options in parameters

Definition at line 392 of file io.c.

References input_opt::change_move_limit, FALSE, input_opt::generate_report, input_opt::have_-input, input_opt::inputfile, input_opt::key_eval_percent, input_opt::middle_op, input_opt::outfile, input_opt::paranoid_leve, input_opt::print_iter, input_opt::save_output, input_opt::tabu_-iterations, input_opt::tabu_list_length, input_opt::tabu_max_decrease, and TRUE.

Referenced by main().

4.3.3.8 void print_end (final_report ∗ *report*, FILE ∗ *file*)

Print final report of Tabu Search

Parameters:

report Report generator

file File to print

Definition at line 741 of file io.c.

References final_report::change_left_count, final_report::change_right_count, final_report::clock_-global, final_report::clock_left, final_report::clock_right, final_report::end_global, final_-report::end_left, final_report::end_right, final_report::init_global, final_report::init_left, final_-report::init_right, best_result::key, final_report::left, final_report::left_iter, final_report::restart_-left_control, final_report::restart_right_control, final_report::right, final_report::right_iter, and best_result::value.

Referenced by open_report().

4.3.3.9 void print_end_test_matrix (final_report ∗ *report*, FILE ∗ *file*)

Print each step Tabu Search for test matrix

Parameters:

report Report generator

file File to print

Definition at line 718 of file io.c.

References final_report::clock_left, final_report::clock_right, final_report::end_left, final_-report::end_right, final_report::init_left, final_report::init_right, best_result::key, final_report::left, final_report::left_iter, final_report::right, final_report::right_iter, and best_result::value.

Referenced by report_use_test_matrix().

4.3.3.10 void print_full_test_options (char ∗ *name*)

Print full test usage and options

Parameters:

name Name for executable

Definition at line 500 of file io.c.

Referenced by main().

4.3.3.11 void print_generate_options (char ∗ *name*)

Print generate usage and options

Parameters:

name Name for executable

Definition at line 573 of file io.c.

Referenced by main(), and parse_generate_arguments().

4.3.3.12 void print_init (ts_params ∗ *params*, input_options ∗ *options*, FILE ∗ *file*)

Print initial asignation for Tabu Search

Parameters:

params Paramenter of Tabu Search

options Tabu Search general input options

file File to print

Definition at line 683 of file io.c.

References tabu_search_params::change_move_limit, FALSE, input_opt::inputfile, tabu_search_-params::key_eval_percent, input_opt::outfile, input_opt::save_output, tabu_search_params::tabu_-iterations, tabu_search_params::tabu_list_length, and tabu_search_params::tabu_max_decrease.

Referenced by open_report().

4.3.3.13 void print_init_test_matrix (ts_params ∗ *params*, input_options ∗ *options*, FILE ∗ *file*)

Print initial asignation for Tabu Search in output text matrix

Parameters:

params Paramenter of Tabu Search

options Tabu Search general input options

file File to print

Definition at line 709 of file io.c.

References tabu_search_params::key_eval_percent, tabu_search_params::tabu_iterations, tabu_-search_params::tabu_list_length, and tabu_search_params::tabu_max_decrease.

Referenced by report_use_test_matrix().

4.3.3.14 void print_iteration (best_result ∗ *best*, int *iter*, int *block*, FILE ∗ *file*)

Print iteration step values

Parameters:

best Best result until this iteration

iter Iteration number

block Block for this iteration {LEFT,RIGHT}

file File to print

Definition at line 781 of file io.c.

References best_result::key, and best_result::value.

Referenced by open_report().

4.3.3.15 void print_middle (unsigned long *key*, float *score*, unsigned int *bit*, int *block*, int *tabuname*, FILE ∗ *file*)

Print step by step middle operation for Tabu Search

Parameters:

key Key in given step

score Score for given step

bit Bit changed in this step

block Block for this step {LEFT,RIGHT}

Definition at line 770 of file io.c.

Referenced by open_report().

4.3.3.16 void print_mold_test_matrix (FILE ∗ *file*)

Print initial significant keyswords for test matrix

Parameters:

file File to print

Definition at line 732 of file io.c.

Referenced by main().

4.3.3.17 void print_paranoid (paranoid ∗ *paranoid*, FILE ∗ *file*)

Print really paranoid step debug, include key movement values, and tabu list

Parameters:

paranoid Paranoid data

file File to print

Definition at line 790 of file io.c.

References tabu_list::bit_position, paranoid_level::block, tabu_list::key, paranoid_level::key, paranoid_level::list, tabu_list::name, tabu_list::next, paranoid_level::tabu, and tabu_list::value.

Referenced by open_report().

4.3.3.18 void print_paranoid_all (paranoid ∗ *paranoid*, FILE ∗ *file*)

Print really paranoid step debug for evaluation function, include block key generation

Parameters:

paranoid Paranoid data

file File to print

Definition at line 826 of file io.c.

References paranoid_level::bit, paranoid_level::block, paranoid_level::evalkey, and paranoid_-level::key.

Referenced by open_report().

4.3.3.19 void print_paranoid_eval (paranoid $*$ *paranoid*, FILE $*$ *file*)

Print really paranoid step debug for evaluation function

Parameters:

paranoid Paranoid data

file File to print

Definition at line 817 of file io.c.

References paranoid_level::bit, paranoid_level::element, paranoid_level::ones, paranoid_-level::percent, and paranoid_level::zeros.

Referenced by open_report().

4.3.3.20 void print_paranoid_move (paranoid $*$ *paranoid*, FILE $*$ *file*)

Print really paranoid step debug, include movement middle key values and evaluation value

Parameters:

paranoid Paranoid data

file File to print

Definition at line 808 of file io.c.

References paranoid_level::bit, paranoid_level::block, paranoid_level::key, and paranoid_-level::score.

Referenced by open_report().

4.3.3.21 void print_test_options (char $*$ *name*)

Print cipher test usage and options

Parameters:

name Name for executable

Definition at line 559 of file io.c.

Referenced by main().

4.3.3.22 void print_ts_options (char $*$ *name*)

Print Tabu Search usage and options

Parameters:

name Name for executable

Definition at line 529 of file io.c.

Referenced by main().

4.3.3.23 cipher_cont∗ read_input (char ∗*filename*)

Read input file and put the content into corresponding structure

Parameters:

filename Filename of input file

Definition at line 20 of file io.c.

References container_node::cipher_message, MAX_FILE_LINES, NMALLOC, container_-node::plain_message, and container_node::size_array.

Referenced by main().

4.3.3.24 void report_use_test_matrix (output_report ∗ *report*)

Change report binding to use test matrix

Parameters:

report Report generator

Definition at line 672 of file io.c.

References output_report::print_end, print_end_test_matrix(), output_report::print_init, and print_init_test_matrix().

Referenced by main().

4.3.3.25 void void_print_iteration (best_result ∗ *best*, int *block*, int *iter*, FILE ∗ *fp*)

DO NOTHING!

Parameters:

best Best result until this iteration

iter Iteration number

block Block for this iteration {LEFT,RIGHT}

file File to print

Definition at line 849 of file io.c.

Referenced by open_report().

4.3.3.26 void void_print_middle (unsigned long *key*, float *score*, unsigned int *bit*, int *block*, int *tabuname*, FILE ∗ *fp*)

DO NOTHING!

Parameters:

key Key in given step

score Score for given step

bit Bit changed in this step

block Block for this step {LEFT,RIGHT}

Definition at line 839 of file io.c.

Referenced by open_report().

4.3.3.27 void void_print_paranoid (paranoid ∗ *paranoid*, FILE ∗ *fp*)

DO NOTHING!

Parameters:

paranoid Paranoid data

file File to print

Definition at line 857 of file io.c.

Referenced by open_report().

4.3.3.28 void void_print_paranoid_all (paranoid ∗ *paranoid*, FILE ∗ *fp*)

DO NOTHING!

Parameters:

paranoid Paranoid data

file File to print

Definition at line 881 of file io.c.

Referenced by open_report().

4.3.3.29 void void_print_paranoid_eval (paranoid ∗ *paranoid*, FILE ∗ *fp*)

DO NOTHING!

Parameters:

paranoid Paranoid data

file File to print

Definition at line 873 of file io.c.

Referenced by open_report().

4.3.3.30 void void_print_paranoid_move (paranoid ∗ *paranoid*, FILE ∗ *fp*)

DO NOTHING!

Parameters:

paranoid Paranoid data

file File to print

Definition at line 865 of file io.c.

Referenced by open_report().

4.4 /Users/freddy/Documents/Research_proyects/TEA/code/main.c File Reference

Main Tabu Search program.

```
#include <stdio.h>
#include <stdlib.h>
#include "tea.h"
#include "tabu_search.h"
#include "io.h"
#include "types.h"
```

Functions

- **int main (int argc, char $**$argv)**

4.4.1 Detailed Description

Main Tabu Search program.

Author:

Freddy Mun~oz Ramirez <frmunoz(at)inf.utfsm.cl>

Date:

Autumn 2006
This code can be re-distributed under MIT License

Definition in file main.c.

4.4.2 Function Documentation

4.4.2.1 int main (int *argc*, char ** *argv*)

Main Tabu Search for TEA application execution program.

Parameters:

arcg number of input arguments
argv input argument string

Definition at line 20 of file main.c.

References close_report(), FALSE, input_opt::have_input, input_opt::inputfile, MALLOC, open_-report(), parse_ts_arguments(), print_ts_options(), read_input(), and tabusearch().

4.5 /Users/freddy/Documents/Research_proyects/TEA/code/tabu_-search.c File Reference

Tabu Search functions.

```
#include <stdio.h>
#include <stdlib.h>
#include <limits.h>
#include <time.h>
#include "tabu_search.h"
#include "tea.h"
#include "io.h"
#include "types.h"
```

Functions

- **void move_alpha (unsigned long ∗key, tabu_list ∗list, unsigned long ∗mistakes, unsigned int ∗control, unsigned int ∗founded, best_result ∗best, ts_params ∗params, output_report ∗report, cipher_cont ∗cpmess, int block)**
- **void move_beta (unsigned long ∗key, tabu_list ∗list, unsigned long ∗mistakes, unsigned int ∗control, unsigned int ∗founded, best_result ∗best, ts_params ∗params, output_report ∗report, cipher_cont ∗cpmess, int block)**
- **float evaluate (unsigned long key, cipher_cont ∗cpmess, int block, output_report ∗report, ts_-params ∗params)**
- **void tabusearch (cipher_cont ∗cpmess, input_options ∗options, output_report ∗report)**
- **void generate_initial_solution (unsigned long ∗key)**
- **void restart_left (unsigned long ∗key)**
- **void restart_right (unsigned long ∗key)**
- **ts_params ∗ create_params (input_options ∗options)**
- **tabu_list ∗ create_tabu_list (ts_params ∗params)**
- **void restart_tabu (tabu_list ∗tabu_t, ts_params ∗params)**
- **void free_tabu (tabu_list ∗tabu, unsigned int name)**

4.5.1 Detailed Description

Tabu Search functions.

Author:

Freddy Mun∼oz Ramirez <frmunoz(at)inf.utfsm.cl>

Date:

Autumn 2006
This code can be re-distributed under MIT License

Definition in file tabu_search.c.

4.5.2 Function Documentation

4.5.2.1 ts_params∗ create_params (input_options ∗ *options*)

Create Tabu Search Params from Tabu Search general options

Parameters:

options Tabu Search general options

Definition at line 854 of file tabu_search.c.

References input_opt::change_move_limit, tabu_search_params::change_move_limit, CHANGE_-MOVE_LIMIT, DEFAULT_TABU_CYCLES, DEFAULT_TABU_LIST_LENGTH, DEFAULT_-TABU_RESTART_LIMIT, input_opt::key_eval_percent, tabu_search_params::key_eval_percent, MALLOC, PERCENT_MULTI, input_opt::tabu_iterations, tabu_search_params::tabu_iterations, input_opt::tabu_list_length, tabu_search_params::tabu_list_length, input_opt::tabu_max_-decrease, and tabu_search_params::tabu_max_decrease.

Referenced by tabusearch().

4.5.2.2 tabu_list∗ create_tabu_list (ts_params ∗ *params*)

Create a new empty Tabu list with lenght specified in parameters

Parameters:

params Parameters for this tabu search

Definition at line 912 of file tabu_search.c.

References tabu_list::bit_position, tabu_list::key, MALLOC, tabu_list::name, tabu_list::next, POST_LIMIT, tabu_search_params::tabu_list_length, and tabu_list::value.

Referenced by tabusearch().

4.5.2.3 float evaluate (unsigned long *key*, cipher_cont ∗ *cpmess*, int *block*, output_report ∗ *report*, ts_params ∗ *params*)

In order to accomplish the evalutaion for key generation its necesary to compare the mirror bits from right to left, and select the worst score betwen all comparision of all keys it's means if a key have 3 points and another 2, our best score is 2 with some percent of accuracy.

(remember the best score is the worst one)

also its necesary that the bits be in correlative order that means

10100100001

& 10101011011

———

10100000001 < — this have just one bit of coincidences

10100100001

& 10101011001

———

```
  10100000001
& 01010000110
  -------
  11110000111 <— this have three bit of coincidence
```

take just the equals bits (no matters if 0 or 1), its must be equals and correlatives.

Parameters:

key Key to evaluate

cpmess Ciphers and Plain messages

block Name for this block (LEFT OR RIGHT)

report Report generator

params General parameter for tabu search

Definition at line 340 of file tabu_search.c.

References paranoid_level::bit, paranoid_level::block, delta_tea(), paranoid_level::element, paranoid_level::evalkey, gamma_tea(), paranoid_level::key, tabu_search_params::key_eval_-percent, LEFT, MALLOC, NMALLOC, paranoid_level::ones, output_report::options, input_-opt::paranoid_leve, output_report::pardeb, paranoid_level::percent, output_report::print_-paranoid_all, output_report::print_paranoid_eval, container_node::size_array, and paranoid_-level::zeros.

Referenced by move_alpha(), move_beta(), and tabusearch().

4.5.2.4 void free_tabu (tabu_list $*$ *tabu*, unsigned int *name*)

Free all memory used by tabu list elements

Parameters:

tabu Tabu list

name name for the initial element of tabu list

Definition at line 969 of file tabu_search.c.

References free_tabu(), tabu_list::name, and tabu_list::next.

Referenced by free_tabu().

4.5.2.5 void generate_initial_solution (unsigned long $*$ *key*)

Generate a random initial key solution.

Parameters:

key where to put the new key

Definition at line 780 of file tabu_search.c.

Referenced by tabusearch().

4.5.2.6 void move_alpha (unsigned long ∗ *key*, tabu_list ∗ *list*, unsigned long ∗ *mistakes*, unsigned int ∗ *control*, unsigned int ∗ *founded*, best_result ∗ *best*, ts_params ∗ *params*, output_report ∗ *report*, cipher_cont ∗ *cpmess*, int *block*)

Tabus Search movement alpha: change the ith bit of one 32 bit key block, it also get the value for evaluation function and put the best movement value into tabu list.

first movement:

generate only 32 bits movements for 32 bits block key[1] and key[3] moving with left bits shift XOR.

example:

```
  010101010101
^ 000000000001
  ---------
  010101010100
  000000000001 << 1 = 000000000010
  010101010101
^ 000000000010
  ---------
  010101010111
```

and so on.

Parameters:

key Key for this movement (a 128 bit key in 4 blocks of 32 bits)

list Tabu list for this move, corresponding to current key block

mistakes Mistakes counter tracker

control Control wich element is the next in the tabu list

founded If the element was founded

best The current best result

params General parameter for tabu search

report Report generator

cpmess Ciphers and Plain messages

block name for this block (LEFT OR RIGHT)

Definition at line 48 of file tabu_search.c.

References paranoid_level::bit, tabu_list::bit_position, paranoid_level::block, evaluate(), FALSE, paranoid_level::key, LEFT, tabu_list::next, output_report::options, output_report::par, input_-opt::paranoid_leve, POST_LIMIT, output_report::print_paranoid_move, paranoid_level::score, paranoid_level::tabu, tabu_search_params::tabu_list_length, TEA_DUAL_BLOCK, and TRUE.

Referenced by tabusearch().

4.5.2.7 void move_beta (unsigned long ∗ *key*, tabu_list ∗ *list*, unsigned long ∗ *mistakes*, unsigned int ∗ *control*, unsigned int ∗ *founded*, best_result ∗ *best*, ts_params ∗ *params*, output_report ∗ *report*, cipher_cont ∗ *cpmess*, int *block*)

Tabus Search movement beta: swap ith bit with ith+1 bit of one 32 bit key block, it also get the value for evaluation function and put the best movement value into tabu list.

second movement:

generate only 31 bits movements for 32 bits block key[1] and key[3] swaping different bits.

(if two bits are equals, then no move... bit pairs 00 or 11 generate no move)

example:

010101010101

swap 1

010101010110

010101010101

swap 2

010101010011

and so on.

everything in this movement its like fisrt movement, but the movement it self changes.

Parameters:

key Key for this movement (a 128 bit key in 4 blocks of 32 bits)
list Tabu list for this move, corresponding to current key block
mistakes Mistakes counter tracker
control Control wich element is the next in the tabu list
founded If the element was founded
best The current best result
params General parameter for tabu search
report Report generator
cpmess Ciphers and Plain messages
block Name for this block (LEFT OR RIGHT)

Definition at line 193 of file tabu_search.c.

References paranoid_level::bit, tabu_list::bit_position, paranoid_level::block, evaluate(), FALSE, paranoid_level::key, LEFT, tabu_list::next, output_report::options, output_report::par, input_opt::paranoid_leve, POST_LIMIT, output_report::print_paranoid_move, paranoid_level::score, paranoid_level::tabu, tabu_search_params::tabu_list_length, and TRUE.

Referenced by tabusearch().

4.5.2.8 void restart_left (unsigned long ∗ *key*)

Restart the left key block with a random key

Parameters:

key where to put the new key

Definition at line 808 of file tabu_search.c.

4.5.2.9 void restart_right (unsigned long $*$ *key*)

Restart the right key block with a random key

Parameters:

key where to put the new key block

Definition at line 831 of file tabu_search.c.

4.5.2.10 void restart_tabu (tabu_list $*$ *tabu_t*, ts_params $*$ *params*)

Change all values of Tabu list to empty default values

Parameters:

tabu_t Tabu list

params Parameters for this tabu search

Definition at line 952 of file tabu_search.c.

References tabu_list::bit_position, tabu_list::key, tabu_list::name, tabu_list::next, POST_LIMIT, tabu_search_params::tabu_list_length, and tabu_list::value.

Referenced by tabusearch().

4.5.2.11 void tabusearch (cipher_cont $*$ *cpmess*, input_options $*$ *options*, output_report $*$ *report*)

General Tabu Search, it triggers the movements, and perform each iteration (include convergency if condition)

Parameters:

cpmess Ciphers and Plain messages

options Tabu Search general options

report Report generator

Definition at line 457 of file tabu_search.c.

References best_result::bit, paranoid_level::block, tabu_search_params::change_move_limit, create_params(), create_tabu_list(), evaluate(), FALSE, generate_initial_solution(), best_result::key, paranoid_level::key, LEFT, paranoid_level::list, MALLOC, tabu_movement::move, move_-alpha(), move_beta(), output_report::options, output_report::par, input_opt::paranoid_leve, output_report::print_init, output_report::print_iteration, output_report::print_paranoid, output_-report::report_file, restart_tabu(), RIGHT, input_opt::save_output, paranoid_level::tabu, TABU_-END_LIMIT, tabu_search_params::tabu_iterations, TRUE, and best_result::value.

Referenced by main().

4.6 /Users/freddy/Documents/Research_proyects/TEA/code/tabu_-search.h File Reference

tabu search constants and function prototypes

```
#include "types.h"
#include <limits.h>
```

Defines

- **#define DEFAULT_TABU_LIST_LENGTH 16**
- **#define DEFAULT_TABU_CYCLES 2000**
- **#define CHANGE_MOVE_LIMIT 2**
- **#define PERCENT_MULTI 0.99**
- **#define DEFAULT_TABU_RESTART_LIMIT 64**
- **#define POST_LIMIT LONG_MAX**
- **#define TEA_DUAL_BLOCK 32**
- **#define TABU_END_LIMIT 32**
- **#define LEFT -1**
- **#define RIGHT 1**

Functions

- **void move_alpha (unsigned long ∗, tabu_list ∗, unsigned long ∗, unsigned int ∗, unsigned int ∗, best_result ∗, ts_params ∗, output_report ∗, cipher_cont ∗, int)**
- **void move_beta (unsigned long ∗, tabu_list ∗, unsigned long ∗, unsigned int ∗, unsigned int ∗, best_result ∗, ts_params ∗, output_report ∗, cipher_cont ∗, int)**
- **float evaluate (unsigned long, cipher_cont ∗, int, output_report ∗, ts_params ∗)**
- **void tabusearch (cipher_cont ∗, input_options ∗, output_report ∗)**
- **void generate_initial_solution (unsigned long ∗)**
- **void restart_left (unsigned long ∗)**
- **void restart_right (unsigned long ∗)**
- **ts_params ∗ create_params (input_options ∗)**
- **tabu_list ∗ create_tabu_list (ts_params ∗)**
- **void restart_tabu (tabu_list ∗, ts_params ∗)**
- **void free_tabu (tabu_list ∗, unsigned int)**

4.6.1 Detailed Description

tabu search constants and function prototypes

Author:

Freddy Mun∼oz Ramirez <frmunoz(at)inf.utfsm.cl>

Date:

Autumn 2006
This code can be re-distributed under MIT License

Definition in file tabu_search.h.

4.6.2 Define Documentation

4.6.2.1 #define CHANGE_MOVE_LIMIT 2

Default Limit for change the movement, the same limit would be used to return to original movement

Definition at line 30 of file tabu_search.h.

Referenced by create_params().

4.6.2.2 #define DEFAULT_TABU_CYCLES 2000

Default number of cycles which perform tabu search

Definition at line 25 of file tabu_search.h.

Referenced by create_params().

4.6.2.3 #define DEFAULT_TABU_LIST_LENGTH 16

Default tabu list lenght

Definition at line 19 of file tabu_search.h.

Referenced by create_params().

4.6.2.4 #define DEFAULT_TABU_RESTART_LIMIT 64

Default number of allowed bad results consecutives before restart.

Definition at line 40 of file tabu_search.h.

Referenced by create_params().

4.6.2.5 #define LEFT -1

Left direction/block constant definition.

Definition at line 64 of file tabu_search.h.

Referenced by evaluate(), move_alpha(), move_beta(), and tabusearch().

4.6.2.6 #define PERCENT_MULTI 0.99

Default percent of bits that the keys in the universe will have in common

Definition at line 35 of file tabu_search.h.

Referenced by create_params().

4.6.2.7 #define POST_LIMIT LONG_MAX

alias for LONG_MAX

Definition at line 46 of file tabu_search.h.

Referenced by create_tabu_list(), move_alpha(), move_beta(), and restart_tabu().

4.6.2.8 #define RIGHT 1

Right direction/block constant definition

Definition at line 69 of file tabu_search.h.

Referenced by tabusearch().

4.6.2.9 #define TABU_END_LIMIT 32

Value that the best result have to reach to finish.

Definition at line 58 of file tabu_search.h.

Referenced by tabusearch().

4.6.2.10 #define TEA_DUAL_BLOCK 32

Size of one key block (a key is composed of 4 key blocks)

Definition at line 52 of file tabu_search.h.

Referenced by move_alpha().

4.6.3 Function Documentation

4.6.3.1 ts_params∗ create_params (input_options ∗ *options*)

Create Tabu Search Params from Tabu Search general options

Parameters:

options Tabu Search general options

Definition at line 854 of file tabu_search.c.

References CHANGE_MOVE_LIMIT, tabu_search_params::change_move_limit, input_-opt::change_move_limit, DEFAULT_TABU_CYCLES, DEFAULT_TABU_LIST_LENGTH, DEFAULT_TABU_RESTART_LIMIT, tabu_search_params::key_eval_percent, input_opt::key_-eval_percent, MALLOC, PERCENT_MULTI, tabu_search_params::tabu_iterations, input_-opt::tabu_iterations, tabu_search_params::tabu_list_length, input_opt::tabu_list_length, tabu_-search_params::tabu_max_decrease, and input_opt::tabu_max_decrease.

Referenced by tabusearch().

4.6.3.2 tabu_list∗ create_tabu_list (ts_params ∗ *params*)

Create a new empty Tabu list with lenght specified in parameters

Parameters:

params Parameters for this tabu search

Definition at line 912 of file tabu_search.c.

References tabu_list::bit_position, tabu_list::key, MALLOC, tabu_list::name, tabu_list::next, POST_LIMIT, tabu_search_params::tabu_list_length, and tabu_list::value.

Referenced by tabusearch().

4.6.3.3 float evaluate (unsigned long *key*, cipher_cont ∗ *cpmess*, int *block*, output_report ∗ *report*, ts_params ∗ *params*)

In order to accomplish the evalutaion for key generation its necesary to compare the mirror bits from right to left, and select the worst score betwen all comparision of all keys it's means if a key have 3 points and another 2, our best score is 2 with some percent of accuracy.

(remember the best score is the worst one)

also its necesary that the bits be in correlative order that means

10100100001

& 10101011011

———

10100000001 < — this have just one bit of coincidences

10100100001

& 10101011001

———

10100000001

& 01010000110

———

11110000111 < — this have three bit of coincidence

take just the equals bits (no matters if 0 or 1), its must be equals and correlatives.

Parameters:

key Key to evaluate

cpmess Ciphers and Plain messages

block Name for this block (LEFT OR RIGHT)

report Report generator

params General parameter for tabu search

Definition at line 340 of file tabu_search.c.

References paranoid_level::bit, paranoid_level::block, delta_tea(), paranoid_level::element, paranoid_level::evalkey, gamma_tea(), paranoid_level::key, tabu_search_params::key_eval_-percent, LEFT, MALLOC, NMALLOC, paranoid_level::ones, output_report::options, input_-opt::paranoid_leve, output_report::pardeb, paranoid_level::percent, output_report::print_-paranoid_all, output_report::print_paranoid_eval, container_node::size_array, and paranoid_-level::zeros.

Referenced by move_alpha(), move_beta(), and tabusearch().

4.6.3.4 void free_tabu (tabu_list ∗ *tabu*, unsigned int *name*)

Free all memory used by tabu list elements

Parameters:

tabu Tabu list

name name for the initial element of tabu list

Definition at line 969 of file tabu_search.c.

References free_tabu(), tabu_list::name, and tabu_list::next.

Referenced by free_tabu().

4.6.3.5 void generate_initial_solution (unsigned long ∗ *key*)

Generate a random initial key solution.

Parameters:

key where to put the new key

Definition at line 780 of file tabu_search.c.

Referenced by tabusearch().

4.6.3.6 void move_alpha (unsigned long ∗ *key*, tabu_list ∗ *list*, unsigned long ∗ *mistakes*, unsigned int ∗ *control*, unsigned int ∗ *founded*, best_result ∗ *best*, ts_params ∗ *params*, output_report ∗ *report*, cipher_cont ∗ *cpmess*, int *block*)

Tabus Search movement alpha: change the ith bit of one 32 bit key block, it also get the value for evaluation function and put the best movement value into tabu list.

first movement:

generate only 32 bits movements for 32 bits block key[1] and key[3] moving with left bits shift XOR.

example:

010101010101

^ 000000000001

010101010100

000000000001 << 1 = 000000000010

010101010101

^ 000000000010

010101010111

and so on.

Parameters:

key Key for this movement (a 128 bit key in 4 blocks of 32 bits)

list Tabu list for this move, corresponding to current key block

mistakes Mistakes counter tracker

control Control wich element is the next in the tabu list

founded If the element was founded

best The current best result

params General parameter for tabu search

report Report generator

cpmess Ciphers and Plain messages

block name for this block (LEFT OR RIGHT)

Definition at line 48 of file tabu_search.c.

References paranoid_level::bit, tabu_list::bit_position, paranoid_level::block, evaluate(), FALSE, paranoid_level::key, LEFT, tabu_list::next, output_report::options, output_report::par, input_-opt::paranoid_leve, POST_LIMIT, output_report::print_paranoid_move, paranoid_level::score, paranoid_level::tabu, tabu_search_params::tabu_list_length, TEA_DUAL_BLOCK, and TRUE.

Referenced by tabusearch().

4.6.3.7 void move_beta (unsigned long ∗ *key*, tabu_list ∗ *list*, unsigned long ∗ *mistakes*, unsigned int ∗ *control*, unsigned int ∗ *founded*, best_result ∗ *best*, ts_params ∗ *params*, output_report ∗ *report*, cipher_cont ∗ *cpmess*, int *block*)

Tabus Search movement beta: swap ith bit with ith+1 bit of one 32 bit key block, it also get the value for evaluation function and put the best movement value into tabu list.

second movement:

generate only 31 bits movements for 32 bits block key[1] and key[3] swaping different bits.

(if two bits are equals, then no move... bit pairs 00 or 11 generate no move)

example:

010101010101

swap 1

010101010110

010101010101

swap 2

010101010011

and so on.

everything in this movement its like fisrt movement, but the movement it self changes.

Parameters:

key Key for this movement (a 128 bit key in 4 blocks of 32 bits)

list Tabu list for this move, corresponding to current key block

mistakes Mistakes counter tracker

control Control wich element is the next in the tabu list

founded If the element was founded

best The current best result

params General parameter for tabu search

report Report generator

cpmess Ciphers and Plain messages

block Name for this block (LEFT OR RIGHT)

Definition at line 193 of file tabu_search.c.

References paranoid_level::bit, tabu_list::bit_position, paranoid_level::block, evaluate(), FALSE, paranoid_level::key, LEFT, tabu_list::next, output_report::options, output_report::par, input_-opt::paranoid_leve, POST_LIMIT, output_report::print_paranoid_move, paranoid_level::score, paranoid_level::tabu, tabu_search_params::tabu_list_length, and TRUE.

Referenced by tabusearch().

4.6.3.8 void restart_left (unsigned long ∗ *key*)

Restart the left key block with a random key

Parameters:

key where to put the new key

Definition at line 808 of file tabu_search.c.

4.6.3.9 void restart_right (unsigned long ∗ *key*)

Restart the right key block with a random key

Parameters:

key where to put the new key block

Definition at line 831 of file tabu_search.c.

4.6.3.10 void restart_tabu (tabu_list ∗ *tabu_t*, ts_params ∗ *params*)

Change all values of Tabu list to empty default values

Parameters:

tabu_t Tabu list

params Parameters for this tabu search

Definition at line 952 of file tabu_search.c.

References tabu_list::bit_position, tabu_list::key, tabu_list::name, tabu_list::next, POST_LIMIT, tabu_search_params::tabu_list_length, and tabu_list::value.

Referenced by tabusearch().

4.6.3.11 void tabusearch (cipher_cont ∗ *cpmess*, input_options ∗ *options*, output_report ∗ *report*)

General Tabu Search, it triggers the movements, and perform each iteration (include convergency if condition)

Parameters:

cpmess Ciphers and Plain messages

options Tabu Search general options

report Report generator

Definition at line 457 of file tabu_search.c.

References best_result::bit, paranoid_level::block, tabu_search_params::change_move_limit, create_params(), create_tabu_list(), evaluate(), FALSE, generate_initial_solution(), paranoid_-level::key, best_result::key, LEFT, paranoid_level::list, MALLOC, tabu_movement::move, move_alpha(), move_beta(), output_report::options, output_report::par, input_opt::paranoid_-leve, output_report::print_init, output_report::print_iteration, output_report::print_paranoid, output_report::report_file, restart_tabu(), RIGHT, input_opt::save_output, paranoid_level::tabu, TABU_END_LIMIT, tabu_search_params::tabu_iterations, TRUE, and best_result::value.

Referenced by main().

4.7 /Users/freddy/Documents/Research_proyects/TEA/code/tea.c File Reference

TEA Implementation.

```
#include "tea.h"
```

Functions

- **void tea_encrypt (unsigned long ∗v, unsigned long ∗w, const unsigned long ∗k)**
- **void tea_decrypt (unsigned long ∗v, unsigned long ∗w, const unsigned long ∗k)**
- **void alpha_tea (unsigned long ∗plain, unsigned long ∗cipher, unsigned long ∗key)**
- **void beta_tea (unsigned long ∗plain, unsigned long ∗cipher, unsigned long ∗key)**
- **void gamma_tea (unsigned long ∗plain, unsigned long ∗cipher, unsigned long ∗key)**
- **void delta_tea (unsigned long ∗plain, unsigned long ∗cipher, unsigned long ∗key)**

4.7.1 Detailed Description

TEA Implementation.

Author:

Freddy Mun∼oz Ramirez <frmunoz(at)inf.utfsm.cl>

Date:

Autumn 2006
This code can be re-distributed under MIT License

Definition in file tea.c.

4.7.2 Function Documentation

4.7.2.1 void alpha_tea (unsigned long ∗ *plain*, unsigned long ∗ *cipher*, unsigned long ∗ *key*)

TEA1 key descomposition for left key block, its calculate the recidual key part (with minus noise) and mold it into key structure a justification for the operation involved in this function can be found in `http://www.inf.utfsm.cl/∼frmunoz/research/tea`

Parameters:

plan Plain Message

cipher Cipher Message

key Key candidate to calculate recidual key part IMPORTANT KEY CANDITATE keyp[0]

Definition at line 53 of file tea.c.

References TEA_DELTA_KEY.

4.7.2.2 void beta_tea (unsigned long ∗ *plain*, unsigned long ∗ *cipher*, unsigned long ∗ *key*)

TEA1 key descomposition for right key block, its calculate the recidual key part (with minus noise) and mold it into key structure a justification for the operation involved in this function can be found in http://www.inf.utfsm.cl/∼frmunoz/research/tea

Parameters:

plan Plain Message

cipher Cipher Message

key Key candidate to calculate recidual key part IMPORTANT KEY CANDITATE keyp[2]

Definition at line 75 of file tea.c.

References TEA_DELTA_KEY.

4.7.2.3 void delta_tea (unsigned long ∗ *plain*, unsigned long ∗ *cipher*, unsigned long ∗ *key*)

TEA1 key descomposition for right key block, its calculate the recidual key part (with minus noise) and mold it into key structure a justification for the operation involved in this function can be found in http://www.inf.utfsm.cl/∼frmunoz/research/tea

Parameters:

plan Plain Message

cipher Cipher Message

key Key candidate to calculate recidual key part IMPORTANT KEY CANDITATE keyp[3]

Definition at line 121 of file tea.c.

References TEA_DELTA_KEY.

Referenced by evaluate().

4.7.2.4 void gamma_tea (unsigned long ∗ *plain*, unsigned long ∗ *cipher*, unsigned long ∗ *key*)

TEA1 key descomposition for left key block, its calculate the recidual key part (with minus noise) and mold it into key structure a justification for the operation involved in this function can be found in http://www.inf.utfsm.cl/∼frmunoz/research/tea

Parameters:

plan Plain Message

cipher Cipher Message

key Key candidate to calculate recidual key part IMPORTANT KEY CANDITATE keyp[1]

Definition at line 97 of file tea.c.

References TEA_DELTA_KEY.

Referenced by evaluate().

4.7.2.5 void tea_decrypt (unsigned long ∗ *v*, unsigned long ∗ *w*, const unsigned long ∗ *k*)

TEA decipher implementation BASE TEA CODE FROM: http://www.simonshepherd.supanet.com/

Parameters:

- ***w*** Cipher message
- ***v*** Where to put plain message
- ***k*** Key

Definition at line 33 of file tea.c.

References TEA_CYCLES, TEA_DELTA_KEY, and TEA_SUM.

Referenced by main().

4.7.2.6 void tea_encrypt (unsigned long ∗ *v*, unsigned long ∗ *w*, const unsigned long ∗ *k*)

TEA cipher implementation BASE TEA CODE FROM: http://www.simonshepherd.supanet.com/sc

Parameters:

- ***v*** Plain message
- ***w*** Where to put cipher message
- ***k*** Key

Definition at line 16 of file tea.c.

References TEA_CYCLES, and TEA_DELTA_KEY.

Referenced by main().

4.8 /Users/freddy/Documents/Research_proyects/TEA/code/tea.h File Reference

TEA constant and functions definitions.

Defines

- `#define` **TEA_CYCLES 1**
- **#define TEA_DELTA_KEY 0x9E3779B9**
- **#define TEA_SUM TEA_DELTA_KEY << 5**

Functions

- `void` **tea_encrypt (unsigned long ∗, unsigned long ∗, const unsigned long ∗)**
- **void tea_decrypt (unsigned long ∗, unsigned long ∗, const unsigned long ∗)**
- **void alpha_tea (unsigned long ∗, unsigned long ∗, unsigned long ∗)**
- **void beta_tea (unsigned long ∗, unsigned long ∗, unsigned long ∗)**
- **void gamma_tea (unsigned long ∗, unsigned long ∗, unsigned long ∗)**
- **void delta_tea (unsigned long ∗, unsigned long ∗, unsigned long ∗)**

4.8.1 Detailed Description

TEA constant and functions definitions.

Author:

Freddy Mun∼oz Ramirez <frmunoz(at)inf.utfsm.cl>

Date:

Autumn 2006
This code can be re-distributed under MIT License

Definition in file **tea.h.**

4.8.2 Define Documentation

4.8.2.1 #define TEA_CYCLES 1

Cycles for TEA

Definition at line 15 of file tea.h.

Referenced by tea_decrypt(), and tea_encrypt().

4.8.2.2 #define TEA_DELTA_KEY 0x9E3779B9

TEA cypher key schedule

Definition at line 20 of file tea.h.

Referenced by alpha_tea(), beta_tea(), delta_tea(), gamma_tea(), tea_decrypt(), and tea_encrypt().

4.8.2.3 #define TEA_SUM TEA_DELTA_KEY << 5

TEA cypher sum

Definition at line 25 of file tea.h.

Referenced by tea_decrypt().

4.8.3 Function Documentation

4.8.3.1 void alpha_tea (unsigned long ∗ *plain*, unsigned long ∗ *cipher*, unsigned long ∗ *key*)

TEA1 key descomposition for left key block, its calculate the recidual key part (with minus noise) and mold it into key structure a justification for the operation involved in this function can be found in `http://www.inf.utfsm.cl/~frmunoz/research/tea`

Parameters:

- ***plan*** Plain Message
- ***cipher*** Cipher Message
- ***key*** Key candidate to calculate recidual key part IMPORTANT KEY CANDITATE keyp[0]

Definition at line 53 of file tea.c.

References TEA_DELTA_KEY.

4.8.3.2 void beta_tea (unsigned long ∗ *plain*, unsigned long ∗ *cipher*, unsigned long ∗ *key*)

TEA1 key descomposition for right key block, its calculate the recidual key part (with minus noise) and mold it into key structure a justification for the operation involved in this function can be found in http://www.inf.utfsm.cl/~frmunoz/research/tea

Parameters:

- ***plan*** Plain Message
- ***cipher*** Cipher Message
- ***key*** Key candidate to calculate recidual key part IMPORTANT KEY CANDITATE keyp[2]

Definition at line 75 of file tea.c.

References TEA_DELTA_KEY.

4.8.3.3 void delta_tea (unsigned long ∗ *plain*, unsigned long ∗ *cipher*, unsigned long ∗ *key*)

TEA1 key descomposition for right key block, its calculate the recidual key part (with minus noise) and mold it into key structure a justification for the operation involved in this function can be found in http://www.inf.utfsm.cl/~frmunoz/research/tea

Parameters:

- ***plan*** Plain Message

cipher Cipher Message

key Key candidate to calculate recidual key part IMPORTANT KEY CANDITATE keyp[3]

Definition at line 121 of file tea.c.

References TEA_DELTA_KEY.

Referenced by evaluate().

4.8.3.4 void gamma_tea (unsigned long ∗ *plain*, unsigned long ∗ *cipher*, unsigned long ∗ *key*)

TEA1 key descomposition for left key block, its calculate the recidual key part (with minus noise) and mold it into key structure a justification for the operation involved in this function can be found in http://www.inf.utfsm.cl/∼frmunoz/research/tea

Parameters:

plan Plain Message

cipher Cipher Message

key Key candidate to calculate recidual key part IMPORTANT KEY CANDITATE keyp[1]

Definition at line 97 of file tea.c.

References TEA_DELTA_KEY.

Referenced by evaluate().

4.8.3.5 void tea_decrypt (unsigned long ∗ *v*, unsigned long ∗ *w*, const unsigned long ∗ *k*)

TEA decipher implementation BASE TEA CODE FROM: http://www.simonshepherd.supanet.com/

Parameters:

w Cipher message

v Where to put plain message

k Key

Definition at line 33 of file tea.c.

References TEA_CYCLES, TEA_DELTA_KEY, and TEA_SUM.

Referenced by main().

4.8.3.6 void tea_encrypt (unsigned long ∗ *v*, unsigned long ∗ *w*, const unsigned long ∗ *k*)

TEA cipher implementation BASE TEA CODE FROM: http://www.simonshepherd.supanet.com/sc

Parameters:

v Plain message

w Where to put cipher message

k `Key`

`Definition at line 16 of file tea.c.`

`References TEA_CYCLES, and TEA_DELTA_KEY.`

`Referenced by main().`

4.9 /Users/freddy/Documents/Research_proyects/TEA/code/test.c File Reference

Test a key in TEA.

```
#include <stdio.h>
#include <stdlib.h>
#include "tea.h"
#include "io.h"
#include "types.h"
```

Functions

- int **main (int argc, char **argv)**

4.9.1 Detailed Description

Test a key in TEA.

Author:

Freddy Mun~oz Ramirez <frmunoz(at)inf.utfsm.cl>

Date:

Autumn 2006
This code can be re-distributed under MIT License

Definition in file **test.c.**

4.9.2 Function Documentation

4.9.2.1 int main (int *argc*, char ** *argv*)

Key test program for TEA cipher. (given a key, it's will test the key over a set of cipher-plain messages)

Parameters:

arcg **number of input arguments**

argv **input argument string**

Definition at line 19 of file test.c.

References FALSE, test_input_opt::have_input, test_input_opt::inputfile, test_input_opt::key1, test_input_opt::key2, test_input_opt::key3, test_input_opt::key4, MALLOC, parse_test_-arguments(), print_test_options(), read_input(), container_node::size_array, tea_decrypt(), and tea_encrypt().

4.10 /Users/freddy/Documents/Research_proyects/TEA/code/ts_-full_test.c File Reference

Full test for tabu search.

```
#include <stdio.h>
#include <stdlib.h>
#include "tea.h"
#include "tabu_search.h"
#include "io.h"
#include "types.h"
#include "ts_full_test.h"
```

Functions

- **int main (int argc, char ∗∗argv)**

4.10.1 Detailed Description

Full test for tabu search.

Author:

Freddy Mun∼oz Ramirez <frmunoz(at)inf.utfsm.cl>

Date:

Autumn 2006
This code can be re-distributed under MIT License

Definition in file ts_full_test.c.

4.10.2 Function Documentation

4.10.2.1 int main (int *argc*, char ∗∗ *argv*)

Generate a test matrix for tabu search over TEA1, varying Tabu Search parameters

Parameters:

arcg **number of input arguments**
argv **input argument string**

Definition at line 23 of file ts_full_test.c.

References input_opt::change_move_limit, close_report(), convert_full_test_opt_to_gen(), FALSE, full_test_input_opt::have_input, full_test_input_opt::init_change_move_limit, full_test_input_-opt::init_tabu_iterations, full_test_input_opt::init_tabu_list_length, full_test_input_opt::init_-tabu_max_decrease, full_test_input_opt::inputfile, input_opt::key_eval_percent, MALLOC,

full_test_input_opt::max_change_move_limit, full_test_input_opt::max_tabu_iterations, full_test_-input_opt::max_tabu_list_length, full_test_input_opt::max_tabu_max_decrease, full_test_input_-opt::min_key_eval_percent, open_report(), parse_full_test_arguments(), print_full_test_options(), print_mold_test_matrix(), read_input(), output_report::report_file, report_use_test_matrix(), input_opt::save_output, input_opt::tabu_iterations, input_opt::tabu_list_length, input_opt::tabu_-max_decrease, tabusearch(), TRUE, TS_CHM_INCREMENT, TS_CHM_MAX, TS_INIT_CHM, TS_INIT_ITER, TS_INIT_LIST, TS_INIT_MAX_DEC, TS_INIT_PERCENT, TS_ITER_-INCREMENT, TS_ITER_MAX, TS_LIS_MAX, TS_LIST_INCREMENT, TS_MAX_DEC_-INCREMENT, TS_MAX_DEC_MAX, TS_PERCENT_INCREMENT, full_test_input_opt::var_-change_move_limit, full_test_input_opt::var_key_eval_percent, full_test_input_opt::var_tabu_-iterations, full_test_input_opt::var_tabu_list_length, and full_test_input_opt::var_tabu_max_-decrease.

4.11 /Users/freddy/Documents/Research_proyects/TEA/code/ts_-full_test.h File Reference

Tabu Search full test constants.

Defines

- #define TS_LIST_INCREMENT 2
- #define TS_ITER_INCREMENT 100
- #define TS_PERCENT_INCREMENT 5
- #define TS_MAX_DEC_INCREMENT 1
- #define TS_CHM_INCREMENT 1
- #define TS_LIS_MAX 31
- #define TS_ITER_MAX 5000
- #define TS_MAX_DEC_MAX 64
- #define TS_CHM_MAX 4
- #define TS_INIT_LIST 2
- #define TS_INIT_ITER 1000
- #define TS_INIT_MAX_DEC 64
- #define TS_INIT_CHM 4
- #define TS_INIT_PERCENT 10

4.11.1 Detailed Description

Tabu Search full test constants.

Author:

Freddy Mun∼oz Ramirez <frmunoz(at)inf.utfsm.cl>

Date:

Autumn 2006
This code can be re-distributed under MIT License

Definition in file ts_full_test.h.

4.11.2 Define Documentation

4.11.2.1 #define TS_CHM_INCREMENT 1

Tabu search change movement increment

Definition at line 30 of file ts_full_test.h.

Referenced by main().

4.11.2.2 #define TS_CHM_MAX 4

Tabu search max change movement limit

Definition at line 46 of file ts_full_test.h.

Referenced by main().

4.11.2.3 #define TS_INIT_CHM 4

Tabu search initial change movement limit

Definition at line 62 of file ts_full_test.h.

Referenced by main().

4.11.2.4 #define TS_INIT_ITER 1000

Tabu search initial iterations

Definition at line 54 of file ts_full_test.h.

Referenced by main().

4.11.2.5 #define TS_INIT_LIST 2

Tabu list initial lenght

Definition at line 50 of file ts_full_test.h.

Referenced by main().

4.11.2.6 #define TS_INIT_MAX_DEC 64

Tabu search initial performance decrement limit

Definition at line 58 of file ts_full_test.h.

Referenced by main().

4.11.2.7 #define TS_INIT_PERCENT 10

Tabu search initial key evaluation percent

Definition at line 66 of file ts_full_test.h.

Referenced by main().

4.11.2.8 #define TS_ITER_INCREMENT 100

Tabu search iteration increment

Definition at line 18 of file ts_full_test.h.

Referenced by main().

4.11.2.9 #define TS_ITER_MAX 5000

Tabu search max iteration number

Definition at line 38 of file ts_full_test.h.

Referenced by main().

4.11.2.10 #define TS_LIS_MAX 31

Tabu search max list lenght number

Definition at line 34 of file ts_full_test.h.

Referenced by main().

4.11.2.11 #define TS_LIST_INCREMENT 2

Tabu list lenght increment

Definition at line 14 of file ts_full_test.h.

Referenced by main().

4.11.2.12 #define TS_MAX_DEC_INCREMENT 1

Tabu search decrement performance increment

Definition at line 26 of file ts_full_test.h.

Referenced by main().

4.11.2.13 #define TS_MAX_DEC_MAX 64

Tabu search max performance decremente

Definition at line 42 of file ts_full_test.h.

Referenced by main().

4.11.2.14 #define TS_PERCENT_INCREMENT 5

Tabu search evaluation key percent increment

Definition at line 22 of file ts_full_test.h.

Referenced by main().

4.12 /Users/freddy/Documents/Research_proyects/TEA/code/types.h File Reference

types definition for all components of TS application

```
#include <time.h>
```

Data Structures

- **struct container_node**

  input data structure Data structure that contains the input data (plain, cipher)

- **struct input_opt**

  Tabu search input options General Tabu search input options data structure.

- **struct test_input_opt**

  Cipher Test input options Cipher Test input options data.

- **struct full_test_input_opt**

  Structure that support input options for full test tabu search input options for full test tabu search.

- **struct generate_options**

  generate plain-cipher input options Container for options in generation of random plain-cipher

- **struct tabu_list**

  Tabu List implemetation data structure The tabu list, its a circular linked list which contains forbidden movements.

- **struct best_result**

  Tabu Search best result implementation Best result for tabu search.

- **struct tabu_search_params**

  Tabu Search parameters A set of parameters to perform tabu search.

- **struct final_report**

  Final report for tabu search Structure that support data for final report generation.

- **struct paranoid_level**

  Paranoid debug support Structure that support paranoid debug for tabu search operations, in order to work each attribute has a different meaning for each different context.

- **struct output_report**

  Structure to support report and debug generation Support for report generation (initial al final report), and support for debugging options, function pointer aids change the function for each different context.

- **struct tabu_movement**

  Structure to support tabu search morphic movement Function pointer aids to support multiple tabu search movements, this structure provides the mechanism to link movement and tabu search call.

Defines

- **#define TRUE 1**
- **#define FALSE 0**
- **#define MALLOC(type) (type ∗)malloc(sizeof(type))**
- **#define NMALLOC(n, type) (type ∗)malloc(n∗sizeof(type))**

Typedefs

- **typedef container_node cipher_cont**

  alias for container_node (p. 7)

- **typedef input_opt input_options**

  alias for input_opt (p. 18) structure

- **typedef test_input_opt test_input_options**

  alias for test_input_opt (p. 30)

- **typedef full_test_input_opt full_test_input_options**

  alias for full_test_input_opt (p. 12)

- **typedef generate_options generate_options**

  alias for generate_options (p. 16)

- **typedef tabu_list tabu_list**

  alias for tabu_list (p. 25)

- **typedef best_result best_result**

  alias for best_result (p. 5)

- **typedef tabu_search_params ts_params**

  alias for tabu_search_params (p. 28)

- **typedef final_report final_report**

  alias for final_report (p. 8)

- **typedef paranoid_level paranoid**

  alias for paranoid_level (p. 24)

- **typedef output_report output_report**

  alias for output_report (p. 21)

- **typedef tabu_movement movement**

  alias for tabu_movement (p. 27)

4.12.1 Detailed Description

types definition for all components of TS application

Author:

Freddy Mun∼oz Ramirez <frmunoz(at)inf.utfsm.cl>

Date:

Autumn 2006
This code can be re-distributed under MIT Licence

Definition in file types.h.

4.12.2 Define Documentation

4.12.2.1 #define FALSE 0

False constant definition.

Definition at line 21 of file types.h.

Referenced by convert_full_test_opt_to_gen(), main(), move_alpha(), move_beta(), parse_full_test_-arguments(), parse_generate_arguments(), parse_test_arguments(), parse_ts_arguments(), print_-init(), and tabusearch().

4.12.2.2 #define MALLOC(type) (type ∗)malloc(sizeof(type))

Memory allocation of size sizeof(type).

Definition at line 26 of file types.h.

Referenced by convert_full_test_opt_to_gen(), create_params(), create_tabu_list(), evaluate(), main(), open_report(), and tabusearch().

4.12.2.3 #define NMALLOC(n, type) (type ∗)malloc(n∗sizeof(type))

Memory allocation of size n∗sizeof(type).

Definition at line 31 of file types.h.

Referenced by evaluate(), and read_input().

4.12.2.4 #define TRUE 1

True constant definition.

Definition at line 16 of file types.h.

Referenced by convert_full_test_opt_to_gen(), main(), move_alpha(), move_beta(), parse_full_-test_arguments(), parse_generate_arguments(), parse_test_arguments(), parse_ts_arguments(), and tabusearch().

Index