SCSP CSP Solver

Generated by Doxygen 1.8.3.1

Fri Apr 19 2013 18:24:19

Contents

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 active_tuple Struct Reference

```
#include <active_tuple.h>
```

Public Attributes

- int pos
- int total_vars
- int ∗ var_map
- int ∗ tuple
- int ∗ tuple_counts
- int ∗ vars
- rb_tree ∗ all_ac_vars
- int all_ac_vars_count
- char ∗∗ var_names
- tree_map ∗ tm
- tree_maps ∗ tms
- llist ∗ added_tuples
- llist ∗ ac_vars_from_tuple
- llist ∗ common_cols
- struct active_tuple ∗ next
- struct active_tuple ∗ last
- int total_added
- int special_generating_solutions
- struct decomposed_tree ∗ treen

3.1.1 Member Data Documentation

3.1.1.1 llist∗ active_tuple::ac_vars_from_tuple

3.1.1.2 llist∗ active_tuple::added_tuples

3.1.1.3 rb_tree∗ active_tuple::all_ac_vars

3.1.1.4 int active_tuple::all_ac_vars_count

3.1.1.5 llist∗ active_tuple::common_cols

3.1.1.6 struct **active_tuple**∗ active_tuple::last

3.1.1.7 struct **active_tuple**∗ active_tuple::next

3.1.1.8 int active_tuple::pos

3.1.1.9 int active_tuple::special_generating_solutions

3.1.1.10 **tree_map**∗ active_tuple::tm

3.1.1.11 **tree_maps**∗ active_tuple::tms

3.1.1.12 int active_tuple::total_added

3.1.1.13 int active_tuple::total_vars

3.1.1.14 struct **decomposed_tree**∗ active_tuple::treen

3.1.1.15 int∗ active_tuple::tuple

3.1.1.16 int∗ active_tuple::tuple_counts

3.1.1.17 int∗ active_tuple::var_map

3.1.1.18 char∗∗ active_tuple::var_names

3.1.1.19 int∗ active_tuple::vars

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

- active_tuple.h

3.2 btd_data Struct Reference

```
#include <btdallsol.h>
```

Public Attributes

- llist ∗ uninit_vars
- llist ∗ init_vars
- active_tuple ∗ goods
- active_tuple ∗ nogoods

3.2.1 Member Data Documentation

3.2.1.1 **active_tuple**∗ btd_data::goods

3.2.1.2 **llist**∗ btd_data::init_vars

3.2.1.3 **active_tuple**∗ btd_data::nogoods

3.2.1.4 **llist**∗ btd_data::uninit_vars

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

- btdallsol.h

3.3 btd_engine Struct Reference

```
#include <btdallsol.h>
```

Public Attributes

- decomposed_tree ∗ tree_root
- decomposed_tree ∗ current_cluster
- llist_node ∗ next_variable_in_current_cluster_to_select
- decomposed_tree ∗∗ variable_highest_cluster

3.3.1 Member Data Documentation

3.3.1.1 decomposed_tree∗ btd_engine::current_cluster

3.3.1.2 llist_node∗ btd_engine::next_variable_in_current_cluster_to_select

3.3.1.3 decomposed_tree∗ btd_engine::tree_root

3.3.1.4 decomposed_tree∗∗ btd_engine::variable_highest_cluster

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

- btdallsol.h

3.4 circular_array Struct Reference

```
#include <circular_array.h>
```

Public Attributes

- int i
- int size
- int ∗ array

3.4.1 Member Data Documentation

3.4.1.1 int∗ circular_array::array

3.4.1.2 int circular_array::i

3.4.1.3 int circular_array::size

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

- circular_array.h

3.5 col_card Struct Reference

```
#include <hashtable.h>
```

Public Attributes

- char ∗ col_name
- int card

3.5.1 Member Data Documentation

3.5.1.1 int col_card::card

3.5.1.2 char∗ col_card::col_name

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

- hashtable.h

3.6 comb_cdata Struct Reference

```
#include <rs2.h>
```

Public Attributes

- llist ∗ comb
- comb_data ∗ cd
- int context

3.6.1 Member Data Documentation

3.6.1.1 comb_data∗ comb_cdata::cd

3.6.1.2 llist∗ comb_cdata::comb

3.6.1.3 int comb_cdata::context

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

- rs2.h

3.7 common_var Struct Reference

```
#include <hashtable.h>
```

Public Attributes

- int apos
- int bpos

3.7.1 Member Data Documentation

3.7.1.1 int common_var::apos

3.7.1.2 int common_var::bpos

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

- hashtable.h

3.8 constraint Struct Reference

```
#include <constraints.h>
```

Public Attributes

- char ∗ name
- int function
- int constraint_id
- parameter ∗∗ parameters
- tuple ∗ tuples
- int ∗ variables
- llist_node ∗∗∗ ac3_1_last
- int uninit_vars
- llist ∗ qgn_references
- char decomposition_added
- struct htable ∗ ht
- int fill
- int lastdftid

3.8.1 Member Data Documentation

3.8.1.1 llist_node∗∗∗ constraint::ac3_1_last

3.8.1.2 int constraint::constraint_id

3.8.1.3 char constraint::decomposition_added

3.8.1.4 int constraint::fill

3.8.1.5 int constraint::function

3.8.1.6 struct htable∗ constraint::ht

3.8.1.7 int constraint::lastdftid

3.8.1.8 char∗ constraint::name

3.8.1.9 parameter∗∗ constraint::parameters

3.8.1.10 llist∗ constraint::qgn_references

3.8.1.11 tuple∗ constraint::tuples

3.8.1.12 int constraint::uninit_vars

3.8.1.13 int∗ constraint::variables

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

- constraints.h

3.9 constraint_graph Struct Reference

```
#include <query_graph.h>
```

Public Attributes

- constraint ∗ root
- int constraint_node_count
- int edge_count
- int cgn_i_size
- constraint_graph_node ∗∗ cgn_i
- llist ∗ node_list
- constraint_graph_edge ∗∗ cge_i
- llist ∗ edge_list
- constraint_graph_edge ∗∗∗ matrix
- llist ∗ left_deep_list
- int fill_edges_count
- int max_arity

3.9.1 Member Data Documentation

3.9.1.1 constraint_graph_edge∗∗ constraint_graph::cge_i

3.9.1.2 constraint_graph_node∗∗ constraint_graph::cgn_i

3.9.1.3 int constraint_graph::cgn_i_size

3.9.1.4 int constraint_graph::constraint_node_count

3.9.1.5 int constraint_graph::edge_count

3.9.1.6 llist∗ constraint_graph::edge_list

3.9.1.7 int constraint_graph::fill_edges_count

3.9.1.8 llist∗ constraint_graph::left_deep_list

3.9.1.9 constraint_graph_edge∗∗∗ constraint_graph::matrix

3.9.1.10 int constraint_graph::max_arity

3.9.1.11 llist∗ constraint_graph::node_list

3.9.1.12 constraint∗ constraint_graph::root

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

- query_graph.h

3.10 constraint_graph_edge Struct Reference

```
#include <query_graph.h>
```

Public Attributes

- int id
- constraint_graph_node * node1
- constraint_graph_node * node2
- int weight
- htable * table
- q_node * query_node
- llist * common_vars

3.10.1 Member Data Documentation

3.10.1.1 llist* constraint_graph_edge::common_vars

3.10.1.2 int constraint_graph_edge::id

3.10.1.3 constraint_graph_node* constraint_graph_edge::node1

3.10.1.4 constraint_graph_node* constraint_graph_edge::node2

3.10.1.5 q_node* constraint_graph_edge::query_node

3.10.1.6 htable* constraint_graph_edge::table

3.10.1.7 int constraint_graph_edge::weight

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

- query_graph.h

3.11 constraint_graph_node Struct Reference

```
#include <query_graph.h>
```

Public Attributes

- constraint * constr
- rbt_set * neighbours
- int color
- char * name
- htable * hash_table
- int table_changed_after_suspend
- int id
- int generation
- int pgeneration

- int combinations_removed
- llist ∗ children
- llist ∗ parents
- int out_of_order
- int peo
- int fill_in
- int heap_pos

3.11.1 Member Data Documentation

3.11.1.1 llist∗ constraint_graph_node::children

3.11.1.2 int constraint_graph_node::color

3.11.1.3 int constraint_graph_node::combinations_removed

3.11.1.4 constraint∗ constraint_graph_node::constr

3.11.1.5 int constraint_graph_node::fill_in

3.11.1.6 int constraint_graph_node::generation

3.11.1.7 htable∗ constraint_graph_node::hash_table

3.11.1.8 int constraint_graph_node::heap_pos

3.11.1.9 int constraint_graph_node::id

3.11.1.10 char∗ constraint_graph_node::name

3.11.1.11 rbt_set∗ constraint_graph_node::neighbours

3.11.1.12 int constraint_graph_node::out_of_order

3.11.1.13 llist∗ constraint_graph_node::parents

3.11.1.14 int constraint_graph_node::peo

3.11.1.15 int constraint_graph_node::pgeneration

3.11.1.16 int constraint_graph_node::table_changed_after_suspend

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

- query_graph.h

3.12 constraint_tree Struct Reference

```
#include <constraints.h>
```

Public Attributes

- int id

- struct constraint_tree_node ∗ root
- int nodes
- int width
- struct constraint_tree_node ∗∗ clusters

3.12.1 Member Data Documentation

3.12.1.1 struct **constraint_tree_node**∗∗ constraint_tree::clusters

3.12.1.2 int constraint_tree::id

3.12.1.3 int constraint_tree::nodes

3.12.1.4 struct **constraint_tree_node**∗ constraint_tree::root

3.12.1.5 int constraint_tree::width

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

- constraints.h

3.13 constraint_tree_node Struct Reference

```
#include <constraints.h>
```

Public Attributes

- int id
- llist ∗ htables
- struct constraint_tree_node ∗ parent
- llist ∗ children

3.13.1 Member Data Documentation

3.13.1.1 **llist**∗ constraint_tree_node::children

3.13.1.2 **llist**∗ constraint_tree_node::htables

3.13.1.3 int constraint_tree_node::id

3.13.1.4 struct **constraint_tree_node**∗ constraint_tree_node::parent

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

- constraints.h

3.14 constraints Struct Reference

```
#include <constraints.h>
```

Public Attributes

- int size
- int count
- constraint ∗ constraint_list
- int max_arity
- int has_intension
- struct rb_tree ∗ tables
- int last_id

3.14.1 Member Data Documentation

3.14.1.1 constraint∗ constraints::constraint_list

3.14.1.2 int constraints::count

3.14.1.3 int constraints::has_intension

3.14.1.4 int constraints::last_id

3.14.1.5 int constraints::max_arity

3.14.1.6 int constraints::size

3.14.1.7 struct rb_tree∗ constraints::tables

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

- constraints.h

3.15 decision_tree Struct Reference

```
#include <oneall.h>
```

Public Attributes

- int id
- double value
- int level
- char ∗ attribute
- struct decision_tree ∗ left
- struct decision_tree ∗ right
- struct decision_tree ∗ parent

3.15.1 Member Data Documentation

3.15.1.1 char∗ decision_tree::attribute

3.15.1.2 int decision_tree::id

3.15.1.3 struct decision_tree∗ decision_tree::left

3.15.1.4 int decision_tree::level

3.15.1.5 struct decision_tree∗ decision_tree::parent

3.15.1.6 struct decision_tree∗ decision_tree::right

3.15.1.7 double decision_tree::value

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

- oneall.h

3.16 decision_tree_system Struct Reference

```
#include <oneall.h>
```

Public Attributes

- char ∗∗ names
- decision_tree ∗ dt

3.16.1 Member Data Documentation

3.16.1.1 decision_tree∗ decision_tree_system::dt

3.16.1.2 char∗∗ decision_tree_system::names

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

- oneall.h

3.17 decomposed_tree Struct Reference

```
#include <decomposition.h>
```

Public Attributes

- int id
- int clique_number
- int dftid
- llist ∗ children
- struct decomposed_tree ∗ parent
- set ∗ vars
- llist ∗ varlist
- set ∗ relation_vars
- set ∗ separator_vars
- llist ∗ constraints
- llist ∗ projected_relations
- llist ∗ comb_datas
- llist ∗ combs
- llist ∗ separator

- llist ∗ separator_var_names
- llist ∗ cluster_var_names
- llist ∗ separator_rels
- llist ∗ projected_rels_on_c_vars
- llist ∗ context_rels
- llist ∗ projected_rels_on_sep_vars
- llist ∗ mini_bucket_rels
- llist ∗ uncovered_vars
- llist ∗ affiliate_rels
- llist ∗∗ affiliates_map
- llist ∗ filter_more
- llist ∗ filter_cluster_relation
- llist ∗ filter_cluster_relation_reverse
- int pre
- int post
- tree_map ∗ parent_separator_tm
- active_tuple ∗ parent_separator_ac
- llist ∗ parent_separator_tm_common_cols
- htable ∗ ancestor_table
- llist ∗ cgs
- set ∗ queue
- llist ∗ fill_edge_constraints
- int level
- int depth
- htable ∗ cluster_relation
- htable ∗ separator_relation
- llist ∗ separator_relations
- struct btd_data ∗ btddata
- llist_node ∗ position_in_parent_node_children_list
- llist ∗ combinations
- int subtree_size
- rb_tree ∗ tables
- rb_tree ∗ f_tables
- rb_tree ∗ ignore_tables
- comb_data ∗∗ representative_comb_datas
- llist ∗ glue_combs
- double weight
- int children_not_processed
- int heap_pos
- int degree
- struct decomposed_tree ∗∗ adjacency
- separator_info ∗ separators
- llist ∗ leafs
- int visit_stamp
- int in_fringe
- rb_tree ∗ stats
- rb_tree ∗ feature_stats
- struct decomposed_tree ∗∗ c_array
- llist ∗ separator_scopes
- int has_stats
- int pending_sep_table_generation

3.17.1 Member Data Documentation

3.17.1.1 struct decomposed_tree∗∗ decomposed_tree::adjacency

3.17.1.2 llist∗ decomposed_tree::affiliate_rels

3.17.1.3 llist∗∗ decomposed_tree::affiliates_map

3.17.1.4 htable∗ decomposed_tree::ancestor_table

3.17.1.5 struct btd_data∗ decomposed_tree::btddata

3.17.1.6 struct decomposed_tree∗∗ decomposed_tree::c_array

3.17.1.7 llist∗ decomposed_tree::cgs

3.17.1.8 llist∗ decomposed_tree::children

3.17.1.9 int decomposed_tree::children_not_processed

3.17.1.10 int decomposed_tree::clique_number

3.17.1.11 htable∗ decomposed_tree::cluster_relation

3.17.1.12 llist∗ decomposed_tree::cluster_var_names

3.17.1.13 llist∗ decomposed_tree::comb_datas

3.17.1.14 llist∗ decomposed_tree::combinations

3.17.1.15 llist∗ decomposed_tree::combs

3.17.1.16 llist∗ decomposed_tree::constraints

3.17.1.17 llist∗ decomposed_tree::context_rels

3.17.1.18 int decomposed_tree::degree

3.17.1.19 int decomposed_tree::depth

3.17.1.20 int decomposed_tree::dftid

3.17.1.21 rb_tree∗ decomposed_tree::f_tables

3.17.1.22 rb_tree∗ decomposed_tree::feature_stats

3.17.1.23 llist∗ decomposed_tree::fill_edge_constraints

3.17.1.24 llist∗ decomposed_tree::filter_cluster_relation

3.17.1.25 llist∗ decomposed_tree::filter_cluster_relation_reverse

3.17.1.26 llist∗ decomposed_tree::filter_more

3.17.1.27 llist∗ decomposed_tree::glue_combs

3.17.1.28 int decomposed_tree::has_stats

3.17.1.29 int decomposed_tree::heap_pos

3.17.1.30 int decomposed_tree::id

3.17.1.31 rb_tree* decomposed_tree::ignore_tables

3.17.1.32 int decomposed_tree::in_fringe

3.17.1.33 llist* decomposed_tree::leafs

3.17.1.34 int decomposed_tree::level

3.17.1.35 llist* decomposed_tree::mini_bucket_rels

3.17.1.36 struct decomposed_tree* decomposed_tree::parent

3.17.1.37 active_tuple* decomposed_tree::parent_separator_ac

3.17.1.38 tree_map* decomposed_tree::parent_separator_tm

3.17.1.39 llist* decomposed_tree::parent_separator_tm_common_cols

3.17.1.40 int decomposed_tree::pending_sep_table_generation

3.17.1.41 llist_node* decomposed_tree::position_in_parent_node_children_list

3.17.1.42 int decomposed_tree::post

3.17.1.43 int decomposed_tree::pre

3.17.1.44 llist* decomposed_tree::projected_relations

3.17.1.45 llist* decomposed_tree::projected_rels_on_c_vars

3.17.1.46 llist* decomposed_tree::projected_rels_on_sep_vars

3.17.1.47 set* decomposed_tree::queue

3.17.1.48 set* decomposed_tree::relation_vars

3.17.1.49 comb_data** decomposed_tree::representative_comb_datas

3.17.1.50 llist* decomposed_tree::separator

3.17.1.51 htable* decomposed_tree::separator_relation

3.17.1.52 llist* decomposed_tree::separator_relations

3.17.1.53 llist* decomposed_tree::separator_rels

3.17.1.54 llist* decomposed_tree::separator_scopes

3.17.1.55 llist* decomposed_tree::separator_var_names

3.17.1.56 set* decomposed_tree::separator_vars

3.17.1.57 separator_info* decomposed_tree::separators

3.17.1.58 rb_tree* decomposed_tree::stats

3.17.1.59 int decomposed_tree::subtree_size

3.17.1.60 rb_tree* decomposed_tree::tables

3.17.1.61 llist* decomposed_tree::uncovered_vars

3.17.1.62 llist* decomposed_tree::varlist

3.17.1.63 set* decomposed_tree::vars

3.17.1.64 int decomposed_tree::visit_stamp

3.17.1.65 double decomposed_tree::weight

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

- decomposition.h

3.18 decomposition_info Struct Reference

```
#include <decomposition.h>
```

Public Attributes

- int number_of_clusters
- int treewidth
- int hyperwidth
- int min_hyperwidth
- int largest_separator_size
- int largest_separator_size_rels
- int mtreewidth
- int mhyperwidth
- int smallest_separator_size
- int smallest_separator_size_rels

3.18.1 Member Data Documentation

3.18.1.1 int decomposition_info::hyperwidth

3.18.1.2 int decomposition_info::largest_separator_size

3.18.1.3 int decomposition_info::largest_separator_size_rels

3.18.1.4 int decomposition_info::mhyperwidth

3.18.1.5 int decomposition_info::min_hyperwidth

3.18.1.6 int decomposition_info::mtreewidth

3.18.1.7 int decomposition_info::number_of_clusters

3.18.1.8 int decomposition_info::smallest_separator_size

3.18.1.9 int decomposition_info::smallest_separator_size_rels

3.18.1.10 int decomposition_info::treewidth

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

- decomposition.h

3.19 gbtd_system Struct Reference

```
#include <gbtd.h>
```

Public Attributes

- pseudo_cluster ∗ root
- pseudo_cluster ∗ current_cluster
- llist ∗ vars
- void ∗ current_var
- options system
- double solCount
- int isolCount
- int total_vars
- int unassigned_vars
- int total_clusters
- int allSol
- set ∗∗ queues
- set ∗ queue
- decomposed_tree ∗∗ treens
- llist_node ∗ current_var_p
- int count_on_m_s
- active_tuple ∗ ac
- comb_data ∗ cd
- int revision_time
- int ∗ tuple_positions
- int ∗ tuple_position_htid
- int wtn

3.19.1 Member Data Documentation

3.19.1.1 active_tuple∗ gbtd_system::ac

3.19.1.2 int gbtd_system::allSol

3.19.1.3 comb_data∗ gbtd_system::cd

3.19.1.4 int gbtd_system::count_on_m_s

3.19.1.5 pseudo_cluster∗ gbtd_system::current_cluster

3.19.1.6 void∗ gbtd_system::current_var

3.19.1.7 llist_node∗ gbtd_system::current_var_p

3.19.1.8 int gbtd_system::isolCount

3.19.1.9 set∗ gbtd_system::queue

3.19.1.10 set∗∗ gbtd_system::queues

3.19.1.11 int gbtd_system::revision_time

3.19.1.12 pseudo_cluster∗ gbtd_system::root

3.19.1.13 double gbtd_system::solCount

3.19.1.14 options gbtd_system::system

3.19.1.15 int gbtd_system::total_clusters

3.19.1.16 int gbtd_system::total_vars

3.19.1.17 decomposed_tree∗∗ gbtd_system::treens

3.19.1.18 int∗ gbtd_system::tuple_position_htid

3.19.1.19 int∗ gbtd_system::tuple_positions

3.19.1.20 int gbtd_system::unassigned_vars

3.19.1.21 llist∗ gbtd_system::vars

3.19.1.22 int gbtd_system::wtn

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

- gbtd.h

3.20 hashmap Struct Reference

```
#include <hashmap.h>
```

Public Attributes

- int id
- int size
- int longest_chain
- int inserted
- llist ∗∗ datap
- int width
- int ∗ col_pos
- int ∗ max

- KEYTYPE ∗ hashmap
- int single
- struct htable ∗ ht
- char ∗ note
- llist ∗ iterator
- llist_node ∗ iteratorptr

3.20.1 Member Data Documentation

3.20.1.1 int∗ hashmap::col_pos

3.20.1.2 llist∗∗ hashmap::datap

3.20.1.3 KEYTYPE∗ hashmap::hashmap

3.20.1.4 struct htable∗ hashmap::ht

3.20.1.5 int hashmap::id

3.20.1.6 int hashmap::inserted

3.20.1.7 llist∗ hashmap::iterator

3.20.1.8 llist_node∗ hashmap::iteratorptr

3.20.1.9 int hashmap::longest_chain

3.20.1.10 int∗ hashmap::max

3.20.1.11 char∗ hashmap::note

3.20.1.12 int hashmap::single

3.20.1.13 int hashmap::size

3.20.1.14 int hashmap::width

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

- hashmap.h

3.21 heap Struct Reference

```
#include <heap.h>
```

Public Attributes

- void ∗∗ body
- int size
- int count

3.21.1 Member Data Documentation

3.21.1.1 void** heap::body

3.21.1.2 int heap::count

3.21.1.3 int heap::size

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

- heap.h

3.22 histogram Struct Reference

```
#include <hashtable.h>
```

Public Attributes

- int columns
- int * values
- int * min
- int * max
- int * dist
- int ** histo

3.22.1 Member Data Documentation

3.22.1.1 int histogram::columns

3.22.1.2 int* histogram::dist

3.22.1.3 int** histogram::histo

3.22.1.4 int* histogram::max

3.22.1.5 int* histogram::min

3.22.1.6 int* histogram::values

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

- hashtable.h

3.23 htable Struct Reference

```
#include <hashtable.h>
```

Public Attributes

- int capacity
- int total_hight

- int last_pos
- int width
- int id
- int tmpid
- char ∗ name
- char ∗ short_name
- int is_first
- int join_cutoff
- char ∗∗ col_names
- llist ∗ col_names_l
- int ∗ vars
- int clustered1
- int clustered2
- int ∗∗ table
- llist ∗ tuple_list
- llist_node ∗∗ tuple_list_array
- llist ∗∗ tuple_pointed_at_from
- llist ∗∗ supports
- llist_node ∗∗ deleted_time_node
- llist ∗ deleted_times
- struct htable ∗ next
- struct htable ∗ prev
- struct htable ∗ last
- struct histogram ∗ histo
- double ∗ entropy
- llist ∗ tree_maps
- int complete
- int incomplete_i
- int incomplete_j
- int incomplete_clustereda
- int incomplete_clusteredb
- int in_tree
- int no_destroy
- constraint ∗ constr
- llist ∗ in_combinations
- llist ∗ in_combinations_hashed
- llist ∗ index_of_ht_in_combinations
- int deleted
- int iterator
- hashmap ∗ htiterator
- int ∗ com_cols_with_next_in_order
- struct tree_node ∗ tn
- struct constraint_graph_node ∗ cgn
- hashmap ∗ jthm
- struct htable ∗ jthmht
- int ∗ jtcolpos
- llist ∗∗ neighbour_list_per_tupple
- int context
- llist ∗ comb_datas
- int tuple_lost
- double ∗ tuple_values
- int problemtable
- int flag
- int markgeneration
- options type
- int assigned

3.23.1 Member Data Documentation

3.23.1.1 int htable::assigned

3.23.1.2 int htable::capacity

3.23.1.3 struct constraint_graph_node∗ htable::cgn

3.23.1.4 int htable::clustered1

3.23.1.5 int htable::clustered2

3.23.1.6 char∗∗ htable::col_names

3.23.1.7 llist∗ htable::col_names_l

3.23.1.8 int∗ htable::com_cols_with_next_in_order

3.23.1.9 llist∗ htable::comb_datas

3.23.1.10 int htable::complete

3.23.1.11 constraint∗ htable::constr

3.23.1.12 int htable::context

3.23.1.13 int htable::deleted

3.23.1.14 llist_node∗∗ htable::deleted_time_node

3.23.1.15 llist∗ htable::deleted_times

3.23.1.16 double∗ htable::entropy

3.23.1.17 int htable::flag

3.23.1.18 struct histogram∗ htable::histo

3.23.1.19 hashmap∗ htable::htiterator

3.23.1.20 int htable::id

3.23.1.21 llist∗ htable::in_combinations

3.23.1.22 llist∗ htable::in_combinations_hashed

3.23.1.23 int htable::in_tree

3.23.1.24 int htable::incomplete_clustereda

3.23.1.25 int htable::incomplete_clusteredb

3.23.1.26 int htable::incomplete_i

3.23.1.27 int htable::incomplete_j

3.23.1.28 llist∗ htable::index_of_ht_in_combinations

3.23.1.29 int htable::is_first

3.23.1.30 int htable::iterator

3.23.1.31 int htable::join_cutoff

3.23.1.32 int∗ htable::jtcolpos

3.23.1.33 hashmap∗ htable::jthm

3.23.1.34 struct htable∗ htable::jthmht

3.23.1.35 struct htable∗ htable::last

3.23.1.36 int htable::last_pos

3.23.1.37 int htable::markgeneration

3.23.1.38 char∗ htable::name

3.23.1.39 llist∗∗ htable::neighbour_list_per_tupple

3.23.1.40 struct htable∗ htable::next

3.23.1.41 int htable::no_destroy

3.23.1.42 struct htable∗ htable::prev

3.23.1.43 int htable::problemtable

3.23.1.44 char∗ htable::short_name

3.23.1.45 llist∗∗ htable::supports

3.23.1.46 int∗∗ htable::table

3.23.1.47 int htable::tmpid

3.23.1.48 struct tree_node∗ htable::tn

3.23.1.49 int htable::total_hight

3.23.1.50 llist∗ htable::tree_maps

3.23.1.51 llist∗ htable::tuple_list

3.23.1.52 llist_node∗∗ htable::tuple_list_array

3.23.1.53 int htable::tuple_lost

3.23.1.54 llist∗∗ htable::tuple_pointed_at_from

3.23.1.55 double∗ htable::tuple_values

3.23.1.56 options htable::type

3.23.1.57 int∗ htable::vars

3.23.1.58 int htable::width

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

- hashtable.h

3.24 htv_undo_data Struct Reference

```
#include <gbtd.h>
```

Public Attributes

- htvar ∗ htv
- llist ∗ undo_deg
- int ∗ min_marks
- llist ∗ list_of_htvar_toundo_domlist_stack

3.24.1 Member Data Documentation

3.24.1.1 htvar∗ htv_undo_data::htv

3.24.1.2 llist∗ htv_undo_data::list_of_htvar_toundo_domlist_stack

3.24.1.3 int∗ htv_undo_data::min_marks

3.24.1.4 llist∗ htv_undo_data::undo_deg

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

- gbtd.h

3.25 htvar Struct Reference

```
#include <gbtd.h>
```

Public Attributes

- htable ∗ ht
- int vid
- int ∗ chosen_value
- int value_lock
- char current_domain_tm_domlist
- llist_node ∗ cd_cursor
- int tm_cursor
- llist ∗ domlist_stack
- int dom_count

3.25.1 Member Data Documentation

3.25.1.1 llist_node∗ htvar::cd_cursor

3.25.1.2 int∗ htvar::chosen_value

3.25.1.3 char htvar::current_domain_tm_domlist

3.25.1.4 int htvar::dom_count

3.25.1.5 llist∗ htvar::domlist_stack

3.25.1.6 htable∗ htvar::ht

3.25.1.7 int htvar::tm_cursor

3.25.1.8 int htvar::value_lock

3.25.1.9 int htvar::vid

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

- gbtd.h

3.26 lastGAC Struct Reference

```
#include <rpwc.h>
```

Public Attributes

- htable ∗ ht
- int pos
- int vpos

3.26.1 Member Data Documentation

3.26.1.1 htable∗ lastGAC::ht

3.26.1.2 int lastGAC::pos

3.26.1.3 int lastGAC::vpos

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

- rpwc.h

3.27 light_stack Struct Reference

```
#include <llist.h>
```

Public Attributes

- struct light_stack_node ∗ head

3.27.1 Member Data Documentation

3.27.1.1 struct **light_stack_node**∗ light_stack::head

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

- llist.h

3.28 light_stack_node Struct Reference

```
#include <llist.h>
```

Public Attributes

- int value
- void ∗ body
- struct light_stack_node ∗ next

3.28.1 Member Data Documentation

3.28.1.1 void∗ light_stack_node::body

3.28.1.2 struct **light_stack_node**∗ light_stack_node::next

3.28.1.3 int light_stack_node::value

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

- llist.h

3.29 llist Struct Reference

```
#include <llist.h>
```

Public Attributes

- struct llist_node ∗ head
- struct llist_node ∗ tail
- struct llist_node ∗ min
- struct llist_node ∗ max
- int id
- int count

3.29.1 Member Data Documentation

3.29.1.1 int llist::count

3.29.1.2 struct **llist_node*** llist::head

3.29.1.3 int llist::id

3.29.1.4 struct **llist_node*** llist::max

3.29.1.5 struct **llist_node*** llist::min

3.29.1.6 struct **llist_node*** llist::tail

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

- llist.h

3.30 llist_node Struct Reference

```
#include <llist.h>
```

Public Attributes

- void * body
- int ibody
- struct llist_node * next
- struct llist_node * previous
- struct llist * list

3.30.1 Member Data Documentation

3.30.1.1 void* llist_node::body

3.30.1.2 int llist_node::ibody

3.30.1.3 struct **llist*** llist_node::list

3.30.1.4 struct **llist_node*** llist_node::next

3.30.1.5 struct **llist_node*** llist_node::previous

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

- llist.h

3.31 main_structure Struct Reference

```
#include <globals.h>
```

Public Attributes

- int valid
- int partition_id
- int function_offset
- variables ∗ vars
- constraints ∗ cons
- llist ∗ domains
- int ∗ expressions
- relation ∗ relations
- int expression_pointer
- int check_counter
- int visit_counter
- int bt_visit_counter
- int bt_counter
- int solve
- int all_solutions
- int init_count
- int consistant_solution_count
- int consistant_print
- int variable_choice_huristic
- int max_domain_value
- int v
- int ∗∗ temp
- int temp_count
- int fc_cbj_fc
- int ac3
- options tree_decom
- options tree_decom_solve
- options tree_decom_solve_1_all
- int tree_decom_wb
- int apply_td
- alive_table ∗ at
- void ∗ tempp
- int left_deep
- int project
- int forward_check
- int use_histo
- int construct_solution
- clock_t split_time
- int only_check_connected
- int total_nodes
- int remaining_nodes
- int db_algo
- double startt
- options r_star_algo
- options r_star_all_single_sol
- options r_star_gac
- int comb_size
- int comb_size_search
- int pprs
- int print_chk
- struct alive_table ∗ alive_t
- struct constraint_graph ∗ cg
- int ∗ vmap

- llist ∗ vmap_dirty_list
- int ∗ cmap
- int rpwc
- int nosingleedge
- llist ∗ effected_vars_by_gac
- int profile
- int m
- int h
- int r
- set ∗ updated_combinations
- int with_index_delete_updating
- int with_tuple_support_recording
- llist ∗ to_queue_supports
- int just_combinations
- int just_redundancy_removal
- int no_tuples
- options no_check_insol_tuples
- struct comb_data ∗ delme
- int remove_redundant_edges
- int ∗ temp_int_p
- char ∗ filename
- int no_support
- int bayesian
- char ∗ tempstr
- int split_cutoff
- int long_tuple_index
- int minibucket
- int minibucketopts
- struct solution ∗ sol
- char ∗ problemname
- int trace_trigger
- long double number_of_solutions
- int comb_ids
- int ht_id
- int tree_bt_time
- options rs2tree_scheme
- options rs2tree_scheme_options
- int rs2tree_scheme_options_parameter
- int feature_vector_options
- llist ∗ active_htables
- llist ∗ tms_list
- int tms_total_inserted
- int dcid
- struct main_structure ∗ next
- struct main_structure ∗ previous
- llist ∗ profile_times
- int constraint_placement
- struct decomposition_info ∗ di
- options queue
- int rel_revise_counter
- options search
- struct decision_tree_system ∗ dts
- int auto_all_revise_count
- int auto_one_revise_count
- int auto_one_ht_revise_count

- int auto_one_ht_revise_count_k
- int auto_all_ht_revise_count
- int auto_all_ht_revise_count_k
- int max_gen_rels
- int max_gen_scope
- int max_gen_tuples
- int max_cluster_tables
- llist ∗ usedhts_list
- int rs2gbtd
- int log
- llist ∗ all_tree_hts
- int node_merge

3.31.1 Member Data Documentation

3.31.1.1 int main_structure::ac3

3.31.1.2 llist∗ main_structure::active_htables

3.31.1.3 struct alive_table∗ main_structure::alive_t

3.31.1.4 int main_structure::all_solutions

3.31.1.5 llist∗ main_structure::all_tree_hts

3.31.1.6 int main_structure::apply_td

3.31.1.7 alive_table∗ main_structure::at

3.31.1.8 int main_structure::auto_all_ht_revise_count

3.31.1.9 int main_structure::auto_all_ht_revise_count_k

3.31.1.10 int main_structure::auto_all_revise_count

3.31.1.11 int main_structure::auto_one_ht_revise_count

3.31.1.12 int main_structure::auto_one_ht_revise_count_k

3.31.1.13 int main_structure::auto_one_revise_count

3.31.1.14 int main_structure::bayesian

3.31.1.15 int main_structure::bt_counter

3.31.1.16 int main_structure::bt_visit_counter

3.31.1.17 struct constraint_graph∗ main_structure::cg

3.31.1.18 int main_structure::check_counter

3.31.1.19 int∗ main_structure::cmap

3.31.1.20 int main_structure::comb_ids

3.31.1.21 int main_structure::comb_size

3.31.1.22 int main_structure::comb_size_search

3.31.1.23 constraints* main_structure::cons

3.31.1.24 int main_structure::consistant_print

3.31.1.25 int main_structure::consistant_solution_count

3.31.1.26 int main_structure::constraint_placement

3.31.1.27 int main_structure::construct_solution

3.31.1.28 int main_structure::db_algo

3.31.1.29 int main_structure::dcid

3.31.1.30 struct comb_data* main_structure::delme

3.31.1.31 struct decomposition_info* main_structure::di

3.31.1.32 llist* main_structure::domains

3.31.1.33 struct decision_tree_system* main_structure::dts

3.31.1.34 llist* main_structure::effected_vars_by_gac

3.31.1.35 int main_structure::expression_pointer

3.31.1.36 int* main_structure::expressions

3.31.1.37 int main_structure::fc_cbj_fc

3.31.1.38 int main_structure::feature_vector_options

3.31.1.39 char* main_structure::filename

3.31.1.40 int main_structure::forward_check

3.31.1.41 int main_structure::function_offset

3.31.1.42 int main_structure::h

3.31.1.43 int main_structure::ht_id

3.31.1.44 int main_structure::init_count

3.31.1.45 int main_structure::just_combinations

3.31.1.46 int main_structure::just_redundancy_removal

3.31.1.47 int main_structure::left_deep

3.31.1.48 int main_structure::log

3.31.1.49 int main_structure::long_tuple_index

3.31.1.50 int main_structure::m

3.31.1.51 int main_structure::max_cluster_tables

3.31.1.52 int main_structure::max_domain_value

3.31.1.53 int main_structure::max_gen_rels

3.31.1.54 int main_structure::max_gen_scope

3.31.1.55 int main_structure::max_gen_tuples

3.31.1.56 int main_structure::minibucket

3.31.1.57 int main_structure::minibucketopts

3.31.1.58 struct main_structure∗ main_structure::next

3.31.1.59 options main_structure::no_check_insol_tuples

3.31.1.60 int main_structure::no_support

3.31.1.61 int main_structure::no_tuples

3.31.1.62 int main_structure::node_merge

3.31.1.63 int main_structure::nosingleedge

3.31.1.64 long double main_structure::number_of_solutions

3.31.1.65 int main_structure::only_check_connected

3.31.1.66 int main_structure::partition_id

3.31.1.67 int main_structure::pprs

3.31.1.68 struct main_structure∗ main_structure::previous

3.31.1.69 int main_structure::print_chk

3.31.1.70 char∗ main_structure::problemname

3.31.1.71 int main_structure::profile

3.31.1.72 llist∗ main_structure::profile_times

3.31.1.73 int main_structure::project

3.31.1.74 options main_structure::queue

3.31.1.75 int main_structure::r

3.31.1.76 options main_structure::r_star_algo

3.31.1.77 options main_structure::r_star_all_single_sol

3.31.1.78 options main_structure::r_star_gac

3.31.1.79 int main_structure::rel_revise_counter

3.31.1.80 relation∗ main_structure::relations

3.31.1.81 int main_structure::remaining_nodes

3.31.1.82 int main_structure::remove_redundant_edges

3.31.1.83 int main_structure::rpwc

3.31.1.84 int main_structure::rs2gbtd

3.31.1.85 options main_structure::rs2tree_scheme

3.31.1.86 options main_structure::rs2tree_scheme_options

3.31.1.87 int main_structure::rs2tree_scheme_options_parameter

3.31.1.88 options main_structure::search

3.31.1.89 struct solution∗ main_structure::sol

3.31.1.90 int main_structure::solve

3.31.1.91 int main_structure::split_cutoff

3.31.1.92 clock_t main_structure::split_time

3.31.1.93 double main_structure::startt

3.31.1.94 int∗∗ main_structure::temp

3.31.1.95 int main_structure::temp_count

3.31.1.96 int∗ main_structure::temp_int_p

3.31.1.97 void∗ main_structure::tempp

3.31.1.98 char∗ main_structure::tempstr

3.31.1.99 llist∗ main_structure::tms_list

3.31.1.100 int main_structure::tms_total_inserted

3.31.1.101 llist∗ main_structure::to_queue_supports

3.31.1.102 int main_structure::total_nodes

3.31.1.103 int main_structure::trace_trigger

3.31.1.104 int main_structure::tree_bt_time

3.31.1.105 options main_structure::tree_decom

3.31.1.106 options main_structure::tree_decom_solve

3.31.1.107 options main_structure::tree_decom_solve_1_all

3.31.1.108 int main_structure::tree_decom_wb

3.31.1.109 set* main_structure::updated_combinations

3.31.1.110 int main_structure::use_histo

3.31.1.111 llist* main_structure::usedhts_list

3.31.1.112 int main_structure::v

3.31.1.113 int main_structure::valid

3.31.1.114 int main_structure::variable_choice_huristic

3.31.1.115 variables* main_structure::vars

3.31.1.116 int main_structure::visit_counter

3.31.1.117 int* main_structure::vmap

3.31.1.118 llist* main_structure::vmap_dirty_list

3.31.1.119 int main_structure::with_index_delete_updating

3.31.1.120 int main_structure::with_tuple_support_recording

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

- globals.h

3.32 node_graph Struct Reference

```
#include <decomposition.h>
```

Public Attributes

- int vid
- llist ** adjacency_list
- char ** matrix
- char out_of_order
- int peo
- int fill_in
- int heap_pos
- int marked_neighbours
- int node_count
- int * node_map

3.32.1 Member Data Documentation

3.32.1.1 llist** node_graph::adjacency_list

3.32.1.2 int node_graph::fill_in

3.32.1.3 int node_graph::heap_pos

3.32.1.4 int node_graph::marked_neighbours

3.32.1.5 char** node_graph::matrix

3.32.1.6 int node_graph::node_count

3.32.1.7 int* node_graph::node_map

3.32.1.8 char node_graph::out_of_order

3.32.1.9 int node_graph::peo

3.32.1.10 int node_graph::vid

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

- decomposition.h

3.33 parameter Struct Reference

```
#include <constraints.h>
```

Public Attributes

- int value_type
- void * value

3.33.1 Member Data Documentation

3.33.1.1 void* parameter::value

3.33.1.2 int parameter::value_type

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

- constraints.h

3.34 pseudo_cluster Struct Reference

```
#include <gbtd.h>
```

Public Attributes

- int id
- int cluster_id
- llist ∗ unassigned_vars
- llist ∗ assigned_vars
- llist ∗ undo_info
- llist ∗ children
- int unassigned_count
- double solCount
- int isolCount
- struct pseudo_cluster ∗ parent
- llist_node ∗ parent_list_self
- int dfs_state
- decomposed_tree ∗ treen
- int all_sol_state
- llist ∗ separator_vars
- tree_map ∗ goods
- tree_map ∗ nogoods
- llist ∗ sep_commons

3.34.1 Member Data Documentation

3.34.1.1 int pseudo_cluster::all_sol_state

3.34.1.2 llist∗ pseudo_cluster::assigned_vars

3.34.1.3 llist∗ pseudo_cluster::children

3.34.1.4 int pseudo_cluster::cluster_id

3.34.1.5 int pseudo_cluster::dfs_state

3.34.1.6 tree_map∗ pseudo_cluster::goods

3.34.1.7 int pseudo_cluster::id

3.34.1.8 int pseudo_cluster::isolCount

3.34.1.9 tree_map∗ pseudo_cluster::nogoods

3.34.1.10 struct pseudo_cluster∗ pseudo_cluster::parent

3.34.1.11 llist_node∗ pseudo_cluster::parent_list_self

3.34.1.12 llist∗ pseudo_cluster::sep_commons

3.34.1.13 llist∗ pseudo_cluster::separator_vars

3.34.1.14 double pseudo_cluster::solCount

3.34.1.15 decomposed_tree∗ pseudo_cluster::treen

3.34.1.16 int pseudo_cluster::unassigned_count

3.34.1.17 llist∗ pseudo_cluster::unassigned_vars

3.34.1.18 llist∗ pseudo_cluster::undo_info

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

- gbtd.h

3.35 rb_tree Struct Reference

```
#include <rb_tree.h>
```

Public Attributes

- int id
- char color
- void ∗ key
- int ikey
- void ∗ payload
- struct rb_tree ∗ parent
- struct rb_tree ∗ left
- struct rb_tree ∗ right

3.35.1 Member Data Documentation

3.35.1.1 char rb_tree::color

3.35.1.2 int rb_tree::id

3.35.1.3 int rb_tree::ikey

3.35.1.4 void∗ rb_tree::key

3.35.1.5 struct rb_tree∗ rb_tree::left

3.35.1.6 struct rb_tree∗ rb_tree::parent

3.35.1.7 void∗ rb_tree::payload

3.35.1.8 struct rb_tree∗ rb_tree::right

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

- rb_tree.h

3.36 rbt_set Struct Reference

```
#include <rbt_set.h>
```

Public Attributes

- int size
- int count
- rb_tree ∗ tmap

3.36.1 Member Data Documentation

3.36.1.1 int rbt_set::count

3.36.1.2 int rbt_set::size

3.36.1.3 rb_tree∗ rbt_set::tmap

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

- rbt_set.h

3.37 rs2tree_configuration Struct Reference

```
#include <rs2tree.h>
```

Public Attributes

- decomposed_tree ∗∗ treens
- llist ∗ usedhts_list
- set ∗ queue
- set ∗∗ queues
- int queues_count

3.37.1 Member Data Documentation

3.37.1.1 set∗ rs2tree_configuration::queue

3.37.1.2 set∗∗ rs2tree_configuration::queues

3.37.1.3 int rs2tree_configuration::queues_count

3.37.1.4 decomposed_tree∗∗ rs2tree_configuration::treens

3.37.1.5 llist∗ rs2tree_configuration::usedhts_list

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

- rs2tree.h

3.38 rs2tree_stats Struct Reference

```
#include <rs2tree.h>
```

Public Attributes

- int id
- llist ∗ varlist
- llist ∗ constraints
- llist ∗ sorted_overlaps
- llist ∗ sorted_overlaps_keys
- llist ∗ all_relations
- int sum0
- int deleted0
- double domain_product0
- double log_domain_sum
- double log_tuples_sum
- int sum1
- int deleted1
- double time
- decomposed_tree ∗ tn
- double f_factor
- double sol_count
- double avg_var_per_rel
- double std_var_per_rel
- double avg_tuple_per_vvp
- double std_tuple_per_vvp
- double avg_tuple_per_vvp_p
- double std_tuple_per_vvp_p
- double avg_tuple_per_vvp_norm
- double std_tuple_per_vvp_norm
- double avg_tuple_per_vvp_norm_p
- double std_tuple_per_vvp_norm_p
- double avg_tuple_per_vvp_log
- double std_tuple_per_vvp_log
- double avg_tuple_per_vvp_log_p
- double std_tuple_per_vvp_log_p
- double entropy_prod_avg
- double entropy_prod_std
- double entropy_sum
- double kappa
- double tuple_vvp_avg
- double tuple_vvp_std
- double tuple_vvp_avg_log
- double tuple_vvp_std_log
- double tuple_vvp_rto
- int relations
- int processed

3.38.1 Member Data Documentation

3.38.1.1 llist∗ rs2tree_stats::all_relations

3.38.1.2 double rs2tree_stats::avg_tuple_per_vvp

3.38.1.3 double rs2tree_stats::avg_tuple_per_vvp_log

3.38.1.4 double rs2tree_stats::avg_tuple_per_vvp_log_p

3.38.1.5 double rs2tree_stats::avg_tuple_per_vvp_norm

3.38.1.6 double rs2tree_stats::avg_tuple_per_vvp_norm_p

3.38.1.7 double rs2tree_stats::avg_tuple_per_vvp_p

3.38.1.8 double rs2tree_stats::avg_var_per_rel

3.38.1.9 llist∗ rs2tree_stats::constraints

3.38.1.10 int rs2tree_stats::deleted0

3.38.1.11 int rs2tree_stats::deleted1

3.38.1.12 double rs2tree_stats::domain_product0

3.38.1.13 double rs2tree_stats::entropy_prod_avg

3.38.1.14 double rs2tree_stats::entropy_prod_std

3.38.1.15 double rs2tree_stats::entropy_sum

3.38.1.16 double rs2tree_stats::f_factor

3.38.1.17 int rs2tree_stats::id

3.38.1.18 double rs2tree_stats::kappa

3.38.1.19 double rs2tree_stats::log_domain_sum

3.38.1.20 double rs2tree_stats::log_tuples_sum

3.38.1.21 int rs2tree_stats::processed

3.38.1.22 int rs2tree_stats::relations

3.38.1.23 double rs2tree_stats::sol_count

3.38.1.24 llist∗ rs2tree_stats::sorted_overlaps

3.38.1.25 llist∗ rs2tree_stats::sorted_overlaps_keys

3.38.1.26 double rs2tree_stats::std_tuple_per_vvp

3.38.1.27 double rs2tree_stats::std_tuple_per_vvp_log

3.38.1.28 double rs2tree_stats::std_tuple_per_vvp_log_p

3.38.1.29 double rs2tree_stats::std_tuple_per_vvp_norm

3.38.1.30 double rs2tree_stats::std_tuple_per_vvp_norm_p

3.38.1.31 double rs2tree_stats::std_tuple_per_vvp_p

3.38.1.32 double rs2tree_stats::std_var_per_rel

3.38.1.33 int rs2tree_stats::sum0

3.38.1.34 int rs2tree_stats::sum1

3.38.1.35 double rs2tree_stats::time

3.38.1.36 decomposed_tree* rs2tree_stats::tn

3.38.1.37 double rs2tree_stats::tuple_vvp_avg

3.38.1.38 double rs2tree_stats::tuple_vvp_avg_log

3.38.1.39 double rs2tree_stats::tuple_vvp_rto

3.38.1.40 double rs2tree_stats::tuple_vvp_std

3.38.1.41 double rs2tree_stats::tuple_vvp_std_log

3.38.1.42 llist* rs2tree_stats::varlist

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

- rs2tree.h

3.39 s_node Struct Reference

```
#include <set.h>
```

Public Attributes

- int key
- void * body

3.39.1 Member Data Documentation

3.39.1.1 void* s_node::body

3.39.1.2 int s_node::key

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

- set.h

3.40 separator_info Struct Reference

```
#include <decomposition.h>
```

Public Attributes

- int rel_count
- htable ** relations

- htable ∗∗ relation_mapped_to
- rb_tree ∗ tables

3.40.1 Member Data Documentation

3.40.1.1 int separator_info::rel_count

3.40.1.2 htable∗∗ separator_info::relation_mapped_to

3.40.1.3 htable∗∗ separator_info::relations

3.40.1.4 rb_tree∗ separator_info::tables

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

- decomposition.h

3.41 set Struct Reference

```
#include <set.h>
```

Public Attributes

- llist ∗ list
- char ∗ map
- llist_node ∗∗ element_ptrs
- int last_removed
- int size
- struct rb_tree ∗ tmap

3.41.1 Member Data Documentation

3.41.1.1 llist_node∗∗ set::element_ptrs

3.41.1.2 int set::last_removed

3.41.1.3 llist∗ set::list

3.41.1.4 char∗ set::map

3.41.1.5 int set::size

3.41.1.6 struct rb_tree∗ set::tmap

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

- set.h

3.42 solution Struct Reference

```
#include <solution.h>
```

Public Attributes

- int ∗ values

3.42.1 Member Data Documentation

3.42.1.1 int∗ solution::values

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

- solution.h

3.43 solution_structure Struct Reference

```
#include <combinations.h>
```

Public Attributes

- int size
- llist ∗∗ solutions

3.43.1 Member Data Documentation

3.43.1.1 int solution_structure::size

3.43.1.2 llist∗∗ solution_structure::solutions

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

- combinations.h

3.44 table_map Struct Reference

```
#include <rs2.h>
```

Public Attributes

- int age
- char ∗∗ map

3.44.1 Member Data Documentation

3.44.1.1 int table_map::age

3.44.1.2 char∗∗ table_map::map

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

- rs2.h

3.45 tree_map Struct Reference

```
#include <tree_map.h>
```

Public Attributes

- llist ∗ indexed_cols
- htable ∗ htto
- llist ∗ lists
- struct tree_map_node ∗ alists
- int id
- int count
- int ccount
- char deleted
- char compressed

3.45.1 Member Data Documentation

3.45.1.1 struct **tree_map_node**∗ tree_map::alists

3.45.1.2 int tree_map::ccount

3.45.1.3 char tree_map::compressed

3.45.1.4 int tree_map::count

3.45.1.5 char tree_map::deleted

3.45.1.6 **htable**∗ tree_map::htto

3.45.1.7 int tree_map::id

3.45.1.8 **llist**∗ tree_map::indexed_cols

3.45.1.9 **llist**∗ tree_map::lists

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

- tree_map.h

3.46 tree_map_node Struct Reference

```
#include <tree_map.h>
```

Public Attributes

- int id
- union {
  llist ∗ lnexts
  struct tree_map_node ∗ anexts
  int ∗∗ atuples
  } nexts

- char full
- int ccount

3.46.1 Member Data Documentation

3.46.1.1 struct tree_map_node∗ tree_map_node::anexts

3.46.1.2 int∗∗ tree_map_node::atuples

3.46.1.3 int tree_map_node::ccount

3.46.1.4 char tree_map_node::full

3.46.1.5 int tree_map_node::id

3.46.1.6 llist∗ tree_map_node::lnexts

3.46.1.7 union { ... } tree_map_node::nexts

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

- tree_map.h

3.47 tree_maps Struct Reference

```
#include <active_tuple.h>
```

Public Attributes

- int trees
- tree_map ∗∗ tms
- int ∗ histogram
- circular_array ∗ ca
- int free
- int ∗ inserted
- int max

3.47.1 Member Data Documentation

3.47.1.1 circular_array∗ tree_maps::ca

3.47.1.2 int tree_maps::free

3.47.1.3 int∗ tree_maps::histogram

3.47.1.4 int∗ tree_maps::inserted

3.47.1.5 int tree_maps::max

3.47.1.6 tree_map∗∗ tree_maps::tms

3.47.1.7 int tree_maps::trees

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

- active_tuple.h

3.48 tripplet Struct Reference

```
#include <rpwc.h>
```

Public Attributes

- int ∗ ttt

3.48.1 Member Data Documentation

3.48.1.1 int∗ tripplet::ttt

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

- rpwc.h

3.49 tuple_ids Struct Reference

```
#include <hashtable.h>
```

Public Attributes

- int ∗ tuple
- int ∗ ids
- int deletet

3.49.1 Member Data Documentation

3.49.1.1 int tuple_ids::deletet

3.49.1.2 int∗ tuple_ids::ids

3.49.1.3 int∗ tuple_ids::tuple

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

- hashtable.h

3.50 undo_info Struct Reference

```
#include <rpwc.h>
```

Public Attributes

- llist ∗∗ reductions
- set ∗ reduction_set
- undo_lastGAC ∗ undo_LG

3.50.1 Member Data Documentation

3.50.1.1 set∗ undo_info::reduction_set

3.50.1.2 llist∗∗ undo_info::reductions

3.50.1.3 undo_lastGAC∗ undo_info::undo_LG

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

- rpwc.h

3.51 undo_lastGAC Struct Reference

```
#include <rpwc.h>
```

Public Attributes

- int ∗∗∗ vars_cons_values
- llist ∗ to_undo

3.51.1 Member Data Documentation

3.51.1.1 llist∗ undo_lastGAC::to_undo

3.51.1.2 int∗∗∗ undo_lastGAC::vars_cons_values

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

- rpwc.h

3.52 value Struct Reference

```
#include <domains.h>
```

Public Attributes

- int value
- int position
- int killed
- int sacremoved
- char ∗ name
- llist_node ∗ domain_list_location

3.52.1 Member Data Documentation

3.52.1.1 llist_node∗ value::domain_list_location

3.52.1.2 int value::killed

3.52.1.3 char∗ value::name

3.52.1.4 int value::position

3.52.1.5 int value::sacremoved

3.52.1.6 int value::value

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

- domains.h

3.53 var_undo_data Struct Reference

```
#include <gbtd.h>
```

Public Attributes

- set ∗ relations
- light_stack ∗ tuples
- light_stack ∗ domains
- variable ∗ var

3.53.1 Member Data Documentation

3.53.1.1 light_stack∗ var_undo_data::domains

3.53.1.2 set∗ var_undo_data::relations

3.53.1.3 light_stack∗ var_undo_data::tuples

3.53.1.4 variable∗ var_undo_data::var

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

- gbtd.h

3.54 var_val Struct Reference

```
#include <variables.h>
```

Public Attributes

- variable ∗ var
- llist_node ∗ value

3.54.1 Member Data Documentation

3.54.1.1 llist_node∗ var_val::value

3.54.1.2 variable∗ var_val::var

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

- variables.h

3.55 var_val_ac Struct Reference

Public Attributes

- variable ∗ var
- llist_node ∗ valn

3.55.1 Member Data Documentation

3.55.1.1 llist_node∗ var_val_ac::valn

3.55.1.2 variable∗ var_val_ac::var

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

- ac.c

3.56 variable Struct Reference

```
#include <variables.h>
```

Public Attributes

- int variable_number
- int domain_size
- value ∗∗ domain
- llist ∗ current_domain
- llist ∗ current_domain_hidden
- llist_node ∗ current_domain_hashmap
- set ∗ current_domain_set
- llist ∗ reductions
- llist_node ∗ choosen_value
- llist ∗ constraints
- int orig_constraint_count
- struct rb_tree ∗ constraints_rb_tree
- llist ∗ htables
- set ∗ neighbours
- llist ∗ future
- llist ∗ past
- llist ∗ removed
- llist ∗ conflicts

- int init_order
- const char ∗ name
- int occurance_count
- int static_order_position
- int max_domain_value
- llist ∗ conflicts_to_remove
- int last_domain_update_time
- llist ∗ htreductions
- int time
- int assigned_count
- struct lastGAC ∗∗∗ last_gac
- struct undo_info ∗ ui

3.56.1 Member Data Documentation

3.56.1.1 int variable::assigned_count

3.56.1.2 llist_node∗ variable::choosen_value

3.56.1.3 llist∗ variable::conflicts

3.56.1.4 llist∗ variable::conflicts_to_remove

3.56.1.5 llist∗ variable::constraints

3.56.1.6 struct rb_tree∗ variable::constraints_rb_tree

3.56.1.7 llist∗ variable::current_domain

3.56.1.8 llist_node∗ variable::current_domain_hashmap

3.56.1.9 llist∗ variable::current_domain_hidden

3.56.1.10 set∗ variable::current_domain_set

3.56.1.11 value∗∗ variable::domain

3.56.1.12 int variable::domain_size

3.56.1.13 llist∗ variable::future

3.56.1.14 llist∗ variable::htables

3.56.1.15 llist∗ variable::htreductions

3.56.1.16 int variable::init_order

3.56.1.17 int variable::last_domain_update_time

3.56.1.18 struct lastGAC∗∗∗ variable::last_gac

3.56.1.19 int variable::max_domain_value

3.56.1.20 const char∗ variable::name

3.56.1.21 set* variable::neighbours

3.56.1.22 int variable::occurance_count

3.56.1.23 int variable::orig_constraint_count

3.56.1.24 llist* variable::past

3.56.1.25 llist* variable::reductions

3.56.1.26 llist* variable::removed

3.56.1.27 int variable::static_order_position

3.56.1.28 int variable::time

3.56.1.29 struct undo_info* variable::ui

3.56.1.30 int variable::variable_number

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

- variables.h

3.57 variables Struct Reference

```
#include <variables.h>
```

Public Attributes

- int count
- int size
- int assigned_count
- llist * variable_list
- llist * variable_list_init
- variable ** map_to_id
- int min_domain
- int max_domain

3.57.1 Member Data Documentation

3.57.1.1 int variables::assigned_count

3.57.1.2 int variables::count

3.57.1.3 variable** variables::map_to_id

3.57.1.4 int variables::max_domain

3.57.1.5 int variables::min_domain

3.57.1.6 int variables::size

3.57.1.7 llist∗ variables::variable_list

3.57.1.8 llist∗ variables::variable_list_init

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

- variables.h

Chapter 4

File Documentation

4.1 ac.c File Reference

```
#include "ac.h"
#include "set.h"
#include "llist.h"
#include "utils.h"
```

Classes

- struct var_val_ac

Typedefs

- typedef struct var_val_ac var_val_ac

Functions

- void ac4 (main_structure ∗m_s, int print, int check)
- int ac3 (int version, main_structure ∗m_s, int print, int check)
- void check_arc_consist (main_structure ∗m_s)

4.1.1 Typedef Documentation

4.1.1.1 typedef struct var_val_ac var_val_ac

4.1.2 Function Documentation

4.1.2.1 int ac3 (int *version*, **main_structure** ∗ *m_s*, int *print*, int *check*)

4.1.2.2 void ac4 (**main_structure** ∗ *m_s*, int *print*, int *check*)

4.1.2.3 void check_arc_consist (**main_structure** ∗ *m_s*)

4.2 ac.h File Reference

```
#include <stdio.h>
#include "globals.h"
#include "variables.h"
#include "revise.h"
```

Functions

- int ac3 (int version, main_structure ∗m_s, int print, int check)
- void ac4 (main_structure ∗m_s, int print, int check)
- void check_arc_consist (main_structure ∗m_s)

4.2.1 Detailed Description

Provides the implementation of arc consistency algorithms.

4.2.2 Function Documentation

4.2.2.1 int ac3 (int *version*, **main_structure** ∗ *m_s*, int *print*, int *check*)

4.2.2.2 void ac4 (**main_structure** ∗ *m_s*, int *print*, int *check*)

4.2.2.3 void check_arc_consist (**main_structure** ∗ *m_s*)

4.3 active_tuple.c File Reference

```
#include "utils.h"
#include "active_tuple.h"
#include "decomposition.h"
#include "hashtable_search.h"
#include "rs2tree.h"
#include "circular_array.h"
#include <math.h>
```

Functions

- tree_maps ∗ new_tree_maps (int trees, llist ∗common_cols)
- void destroy_tree_maps (tree_maps ∗tms)
- active_tuple ∗ new_active_tuple (int number_of_vars, int length, llist ∗lvid, rb_tree ∗all_ac_vars)
- active_tuple ∗ new_active_tuple_sizeable (int number_of_vars, int length, llist ∗lvid, int sizeable, rb_tree ∗all_ac_vars)
- void add_active_tuple (active_tuple ∗ac, int number_of_vars, int length, llist ∗lvid)
- void destroy_active_tuple (active_tuple ∗ac)
- void extend_tuple (active_tuple ∗ac, htable ∗ht, int t)
- void extend_tuple_with_single_value (active_tuple ∗ac, variable ∗var, int value)
- void pop_tuple (active_tuple ∗ac)
- void store_tuple (active_tuple ∗ac, int wildcard)
- void store_tuple_payload (active_tuple ∗ac, int wildcard, void ∗payload)
- void reset_ac_tm (active_tuple ∗ac)

- int get_stored_weight_for_tuple (active_tuple ∗ac, htable ∗ht, int t, long double ∗weight)
- int get_stored_weight_for_tuple_from_tree (active_tuple ∗ac, htable ∗ht, int t, long double ∗weight, tree_map ∗tm)
- void store_weight_for_tuple (active_tuple ∗ac, htable ∗ht, int t, long double weight)

4.3.1 Function Documentation

4.3.1.1 void add_active_tuple (**active_tuple** ∗ *ac,* int *number_of_vars,* int *length,* **llist** ∗ *lvid*)

4.3.1.2 void destroy_active_tuple (**active_tuple** ∗ *ac*)

4.3.1.3 void destroy_tree_maps (**tree_maps** ∗ *tms*)

4.3.1.4 void extend_tuple (**active_tuple** ∗ *ac,* **htable** ∗ *ht,* int *t*)

4.3.1.5 void extend_tuple_with_single_value (**active_tuple** ∗ *ac,* **variable** ∗ *var,* int *value*)

4.3.1.6 int get_stored_weight_for_tuple (**active_tuple** ∗ *ac,* **htable** ∗ *ht,* int *t,* long double ∗ *weight*)

4.3.1.7 int get_stored_weight_for_tuple_from_tree (**active_tuple** ∗ *ac,* **htable** ∗ *ht,* int *t,* long double ∗ *weight,* **tree_map** ∗ *tm*)

4.3.1.8 **active_tuple**∗ new_active_tuple (int *number_of_vars,* int *length,* **llist** ∗ *lvid,* **rb_tree** ∗ *all_ac_vars*)

4.3.1.9 **active_tuple**∗ new_active_tuple_sizeable (int *number_of_vars,* int *length,* **llist** ∗ *lvid,* int *sizeable,* **rb_tree** ∗ *all_ac_vars*)

4.3.1.10 **tree_maps**∗ new_tree_maps (int *trees,* **llist** ∗ *common_cols*)

4.3.1.11 void pop_tuple (**active_tuple** ∗ *ac*)

4.3.1.12 void reset_ac_tm (**active_tuple** ∗ *ac*)

4.3.1.13 void store_tuple (**active_tuple** ∗ *ac,* int *wildcard*)

generating the table

end of creating the table/index

generating the table

end of creating the table/index

4.3.1.14 void store_tuple_payload (**active_tuple** ∗ *ac,* int *wildcard,* void ∗ *payload*)

4.3.1.15 void store_weight_for_tuple (**active_tuple** ∗ *ac,* **htable** ∗ *ht,* int *t,* long double *weight*)

4.4 active_tuple.h File Reference

```
#include "tree_map.h"
#include "circular_array.h"
#include "rb_tree.h"
```

Classes

- struct tree_maps
- struct active_tuple

Typedefs

- typedef struct tree_maps tree_maps
- typedef struct active_tuple active_tuple

Functions

- tree_maps ∗ new_tree_maps (int trees, llist ∗common_cols)
- active_tuple ∗ new_active_tuple_sizeable (int number_of_vars, int length, llist ∗lvid, int sizeable, rb_tree ∗all-_ac_vars)
- active_tuple ∗ new_active_tuple (int number_of_vars, int length, llist ∗lvid, rb_tree ∗all_ac_vars)
- void destroy_active_tuple (active_tuple ∗ac)
- void destroy_tree_maps (tree_maps ∗tms)
- void extend_tuple (active_tuple ∗ac, htable ∗ht, int t)
- void pop_tuple (active_tuple ∗ac)
- void store_tuple_payload (active_tuple ∗ac, int wildcard, void ∗payload)
- void store_tuple (active_tuple ∗ac, int wildcard)
- void extend_tuple_with_single_value (active_tuple ∗ac, variable ∗var, int value)
- void add_active_tuple (active_tuple ∗ac, int number_of_vars, int length, llist ∗lvid)
- void ∗ set_tuple_int_weight (active_tuple ∗ac, int weight)
- int get_stored_weight_for_tuple_from_tree (active_tuple ∗ac, htable ∗ht, int t, long double ∗weight, tree_map ∗tm)
- int get_stored_weight_for_tuple (active_tuple ∗ac, htable ∗ht, int t, long double ∗weight)
- void store_weight_for_tuple (active_tuple ∗ac, htable ∗ht, int t, long double weight)
- void reset_ac_tm (active_tuple ∗ac)

4.4.1 Detailed Description

Provides a tree structure representation of tuples and methods for constructing and querying them.

4.4.2 Typedef Documentation

4.4.2.1 typedef struct **active_tuple active_tuple**

4.4.2.2 typedef struct **tree_maps tree_maps**

4.4.3 Function Documentation

4.4.3.1 void add_active_tuple (**active_tuple** ∗ *ac,* int *number_of_vars,* int *length,* **llist** ∗ *lvid*)

4.4.3.2 void destroy_active_tuple (**active_tuple** ∗ *ac*)

4.4.3.3 void destroy_tree_maps (**tree_maps** ∗ *tms*)

4.4.3.4 void extend_tuple (**active_tuple** ∗ *ac,* **htable** ∗ *ht,* int *t*)

4.4.3.5 void extend_tuple_with_single_value (**active_tuple** ∗ *ac,* **variable** ∗ *var,* int *value*)

4.4.3.6 int get_stored_weight_for_tuple (**active_tuple** ∗ *ac,* **htable** ∗ *ht,* int *t,* long double ∗ *weight*)

4.4.3.7 int get_stored_weight_for_tuple_from_tree (**active_tuple** ∗ *ac,* **htable** ∗ *ht,* int *t,* long double ∗ *weight,* **tree_map** ∗ *tm*)

4.4.3.8 **active_tuple**∗ new_active_tuple (int *number_of_vars,* int *length,* **llist** ∗ *lvid,* **rb_tree** ∗ *all_ac_vars*)

4.4.3.9 **active_tuple**∗ new_active_tuple_sizeable (int *number_of_vars,* int *length,* **llist** ∗ *lvid,* int *sizeable,* **rb_tree** ∗ *all_ac_vars*)

4.4.3.10 **tree_maps**∗ new_tree_maps (int *trees,* **llist** ∗ *common_cols*)

4.4.3.11 void pop_tuple (**active_tuple** ∗ *ac*)

4.4.3.12 void reset_ac_tm (**active_tuple** ∗ *ac*)

4.4.3.13 void∗ set_tuple_int_weight (**active_tuple** ∗ *ac,* int *weight*)

4.4.3.14 void store_tuple (**active_tuple** ∗ *ac,* int *wildcard*)

generating the table

end of creating the table/index

generating the table

end of creating the table/index

4.4.3.15 void store_tuple_payload (**active_tuple** ∗ *ac,* int *wildcard,* void ∗ *payload*)

4.4.3.16 void store_weight_for_tuple (**active_tuple** ∗ *ac,* **htable** ∗ *ht,* int *t,* long double *weight*)

4.5 btdallsol.c File Reference

```
#include "btdallsol.h"
#include "rpwcprocedures.h"
#include "rs2tree.h"
#include "rs2.h"
```

Functions

- int find_match_good_nogood (decomposed_tree ∗child, int goods, long double ∗weight)
- int find_match_good_nogood_from_tree (decomposed_tree ∗child, int goods, long double ∗weight, tree_map ∗tm)
- int update_good_weight_from_tree (decomposed_tree ∗child, tree_map ∗tm, long double weight)
- void update_good_weight (decomposed_tree ∗child, long double weight)
- int btd (btd_engine btde, decomposed_tree ∗root, int recid, long double ∗consistent_sols)
- void init_btd_engine (btd_engine btde, decomposed_tree ∗root)
- void init_btd_tree (decomposed_tree ∗treen)
- btd_data ∗ new_btd_data (decomposed_tree ∗treen)
- btd_data ∗ new_btd_data2 (decomposed_tree ∗treen, int sizeable)
- variable ∗ btd_label (variable ∗var_i, int ∗consistent, long double ∗consistent_count, btd_engine btde, decomposed_tree ∗root, int recid)
- variable ∗ btd_unlabel (variable ∗var_i, int ∗consistant, main_structure ∗m_s)
- variable ∗ choose_variable_from_tree (btd_engine btde)

- void print_separator (decomposed_tree ∗treen, int goods)
- void print_separator_h (decomposed_tree ∗treen, int goods)
- int btd2 (decomposed_tree ∗treen, long double ∗total_sols_ret)
- int btd2_rec (decomposed_tree ∗treen, long double ∗total_sols_ret)
- variable ∗ btd_un_assign_variable (light_stack ∗assigned_var_stack, light_stack ∗undo_value_stack)
- void btd_undo_value_deletions (light_stack ∗value_stack)
- int btd_assign_variable (variable ∗var)
- void btd_prepare_tree_for_btd2 (decomposed_tree ∗treen)
- void btd_reset_variable_domain (variable ∗var)
- void btd_reset_variable_domain2 (variable ∗var)

4.5.1 Function Documentation

4.5.1.1 int btd (**btd_engine** *btde,* **decomposed_tree** ∗ *root,* int *recid,* long double ∗ *consistent_sols*)

4.5.1.2 int btd2 (**decomposed_tree** ∗ *treen,* long double ∗ *total_sols_ret*)

4.5.1.3 int btd2_rec (**decomposed_tree** ∗ *treen,* long double ∗ *total_sols_ret*)

4.5.1.4 int btd_assign_variable (**variable** ∗ *var*)

4.5.1.5 **variable**∗ btd_label (**variable** ∗ *var_i,* int ∗ *consistent,* long double ∗ *consistent_count,* **btd_engine** *btde,* **decomposed_tree** ∗ *root,* int *recid*)

4.5.1.6 void btd_prepare_tree_for_btd2 (**decomposed_tree** ∗ *treen*)

4.5.1.7 void btd_reset_variable_domain (**variable** ∗ *var*)

4.5.1.8 void btd_reset_variable_domain2 (**variable** ∗ *var*)

4.5.1.9 **variable**∗ btd_un_assign_variable (**light_stack** ∗ *assigned_var_stack,* **light_stack** ∗ *undo_value_stack*)

4.5.1.10 void btd_undo_value_deletions (**light_stack** ∗ *value_stack*)

4.5.1.11 **variable**∗ btd_unlabel (**variable** ∗ *var_i,* int ∗ *consistant,* **main_structure** ∗ *m_s*)

4.5.1.12 **variable**∗ choose_variable_from_tree (**btd_engine** *btde*)

4.5.1.13 int find_match_good_nogood (**decomposed_tree** ∗ *child,* int *goods,* long double ∗ *weight*)

4.5.1.14 int find_match_good_nogood_from_tree (**decomposed_tree** ∗ *child,* int *goods,* long double ∗ *weight,* **tree_map** ∗ *tm*)

4.5.1.15 void init_btd_engine (**btd_engine** *btde,* **decomposed_tree** ∗ *root*)

4.5.1.16 void init_btd_tree (**decomposed_tree** ∗ *treen*)

4.5.1.17 **btd_data**∗ new_btd_data (**decomposed_tree** ∗ *treen*)

4.5.1.18 **btd_data**∗ new_btd_data2 (**decomposed_tree** ∗ *treen,* int *sizeable*)

4.5.1.19 void print_separator (**decomposed_tree** ∗ *treen,* int *goods*)

4.5.1.20 void print_separator_h (**decomposed_tree** ∗ *treen,* int *goods*)

4.5.1.21 void update_good_weight (decomposed_tree ∗ *child,* long double *weight*)

4.5.1.22 int update_good_weight_from_tree (decomposed_tree ∗ *child,* tree_map ∗ *tm,* long double *weight*)

4.6 btdallsol.h File Reference

```
#include "utils.h"
#include "llist.h"
#include "active_tuple.h"
#include "decomposition.h"
```

Classes

- struct btd_data
- struct btd_engine

Typedefs

- typedef struct btd_data btd_data
- typedef struct btd_engine btd_engine

Functions

- int find_match_good_nogood_from_tree (decomposed_tree ∗child, int goods, long double ∗weight, tree_map ∗tm)
- int find_match_good_nogood (decomposed_tree ∗child, int goods, long double ∗weight)
- int btd (btd_engine btde, decomposed_tree ∗root, int recid, long double ∗consistent_sols)
- void init_btd_engine (btd_engine btde, decomposed_tree ∗root)
- void init_btd_tree (decomposed_tree ∗treen)
- btd_data ∗ new_btd_data2 (decomposed_tree ∗treen, int sizeable)
- btd_data ∗ new_btd_data (decomposed_tree ∗treen)
- int btd_bcssp (decomposed_tree ∗treen, btd_engine btde)
- variable ∗ btd_label (variable ∗var_i, int ∗consistent, long double ∗consistent_count, btd_engine btde, decomposed_tree ∗root, int recid)
- variable ∗ btd_unlabel (variable ∗var_i, int ∗consistant, main_structure ∗m_s)
- variable ∗ choose_variable_from_tree (btd_engine btde)
- void print_separator (decomposed_tree ∗treen, int goods)
- void print_separator_h (decomposed_tree ∗treen, int goods)
- int btd2_rec (decomposed_tree ∗treen, long double ∗total_sols_ret)
- int btd2 (decomposed_tree ∗treen, long double ∗total_sols_ret)
- variable ∗ btd_un_assign_variable (light_stack ∗assigned_var_stack, light_stack ∗undo_value_stack)
- void btd_undo_value_deletions (light_stack ∗undo_value_stack)
- int btd_assign_variable (variable ∗var)
- void btd_prepare_tree_for_btd2 (decomposed_tree ∗treen)
- void btd_reset_variable_domain (variable ∗var)
- void btd_reset_variable_domain2 (variable ∗var)
- void update_good_weight (decomposed_tree ∗child, long double weight)
- int update_good_weight_from_tree (decomposed_tree ∗child, tree_map ∗tm, long double weight)

4.6.1 Detailed Description

Provides the implementation for finding all solutions using BTD.

4.6.2 Typedef Documentation

4.6.2.1 typedef struct btd_data btd_data

4.6.2.2 typedef struct btd_engine btd_engine

4.6.3 Function Documentation

4.6.3.1 int btd (btd_engine *btde,* decomposed_tree ∗ *root,* int *recid,* long double ∗ *consistent_sols*)

4.6.3.2 int btd2 (decomposed_tree ∗ *treen,* long double ∗ *total_sols_ret*)

4.6.3.3 int btd2_rec (decomposed_tree ∗ *treen,* long double ∗ *total_sols_ret*)

4.6.3.4 int btd_assign_variable (variable ∗ *var*)

4.6.3.5 int btd_bcssp (decomposed_tree ∗ *treen,* btd_engine *btde*)

4.6.3.6 variable∗ btd_label (variable ∗ *var_i,* int ∗ *consistent,* long double ∗ *consistent_count,* btd_engine *btde,* decomposed_tree ∗ *root,* int *recid*)

4.6.3.7 void btd_prepare_tree_for_btd2 (decomposed_tree ∗ *treen*)

4.6.3.8 void btd_reset_variable_domain (variable ∗ *var*)

4.6.3.9 void btd_reset_variable_domain2 (variable ∗ *var*)

4.6.3.10 variable∗ btd_un_assign_variable (light_stack ∗ *assigned_var_stack,* light_stack ∗ *undo_value_stack*)

4.6.3.11 void btd_undo_value_deletions (light_stack ∗ *undo_value_stack*)

4.6.3.12 variable∗ btd_unlabel (variable ∗ *var_i,* int ∗ *consistant,* main_structure ∗ *m_s*)

4.6.3.13 variable∗ choose_variable_from_tree (btd_engine *btde*)

4.6.3.14 int find_match_good_nogood (decomposed_tree ∗ *child,* int *goods,* long double ∗ *weight*)

4.6.3.15 int find_match_good_nogood_from_tree (decomposed_tree ∗ *child,* int *goods,* long double ∗ *weight,* tree_map ∗ *tm*)

4.6.3.16 void init_btd_engine (btd_engine *btde,* decomposed_tree ∗ *root*)

4.6.3.17 void init_btd_tree (decomposed_tree ∗ *treen*)

4.6.3.18 btd_data∗ new_btd_data (decomposed_tree ∗ *treen*)

4.6.3.19 btd_data∗ new_btd_data2 (decomposed_tree ∗ *treen,* int *sizeable*)

4.6.3.20 void print_separator (decomposed_tree ∗ *treen,* int *goods*)

4.6.3.21 void print_separator_h (decomposed_tree ∗ *treen,* int *goods*)

4.6.3.22 void update_good_weight (decomposed_tree ∗ *child,* long double *weight*)

4.6.3.23 int update_good_weight_from_tree (decomposed_tree ∗ *child,* tree_map ∗ *tm,* long double *weight*)

4.7 circular_array.c File Reference

```
#include "utils.h"
#include "circular_array.h"
```

Functions

- circular_array ∗ new_circular_array (int size)
- int add_to_circular_array (int item, circular_array ∗ca)
- void destroy_circular_array (circular_array ∗ca)

4.7.1 Function Documentation

4.7.1.1 int add_to_circular_array (int *item,* **circular_array** ∗ *ca*)

4.7.1.2 void destroy_circular_array (**circular_array** ∗ *ca*)

4.7.1.3 **circular_array**∗ new_circular_array (int *size*)

4.8 circular_array.h File Reference

Classes

- struct circular_array

Typedefs

- typedef struct circular_array circular_array

Functions

- circular_array ∗ new_circular_array (int size)
- int add_to_circular_array (int item, circular_array ∗ca)
- void destroy_circular_array (circular_array ∗ca)

4.8.1 Detailed Description

Provides the implementation of a circular array data-structure.

4.8.2 Typedef Documentation

4.8.2.1 typedef struct **circular_array circular_array**

4.8.3 Function Documentation

4.8.3.1 int add_to_circular_array (int *item,* **circular_array** ∗ *ca*)

4.8.3.2 void destroy_circular_array (**circular_array** ∗ *ca*)

4.8.3.3 circular_array∗ new_circular_array (int *size*)

4.9 combinations.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include "combinations.h"
#include "llist.h"
#include "utils.h"
#include "tree.h"
#include "globals_comb.h"
```

Functions

- int comb (llist ∗∗graphl, int c, int l, int k, set ∗alpha, tree ∗b)
- void collect_solutions (int ∗counts, int size, llist ∗branches, llist ∗solutions_list)
- void collect_solutions_with_all_combs (int ∗counts, int size, llist ∗branches, llist ∗solutions_list)
- void print_combinations (llist ∗combs)
- void print_combination (llist ∗comb)
- int check_constraint (llist ∗combR, llist ∗combL)
- int dac_check_join (int var, int l, llist_node ∗∗assigned_values, llist ∗∗domain_lists, llist ∗∗undo_list, int ∗cc, llist ∗remaining_vars)
- int mac_check_join (int var, int l, llist_node ∗∗assigned_values, llist ∗∗domain_lists, llist ∗∗undo_list, int ∗cc, llist ∗remaining_vars)
- int forward_check_join (int var, int l, llist_node ∗∗assigned_values, llist ∗∗domain_lists, llist ∗∗undo_list, int ∗cc, llist ∗remaining_vars)
- int assign_next_join (int x, llist_node ∗∗assigned_values, llist ∗∗domain_lists, llist ∗∗assigned_current_-domains)
- void undo_filtering_join (int l, llist ∗∗undo_list, llist ∗∗domain_lists)
- void backtrack_search_join (int l, llist ∗∗∗solutions, llist ∗solutions_list)
- int choose_var_join (llist ∗remaining_vars, llist ∗∗domain_lists)
- int backtrack_search_join_rec (int l, llist_node ∗∗assigned_values, llist ∗∗assigned_current_domains, llist ∗remaining_vars, llist ∗∗∗solutions, llist ∗solutions_list, llist ∗∗domain_lists)
- llist ∗∗ get_solutions_of_size (tree ∗n, int k)
- void combine_check_solution (int ∗counts, int size, void ∗arg1, void ∗arg2)
- void count_base_k_function (int ∗xk, int i, int l, int k, llist ∗children, llist ∗solution_list)
- void memo_count_base_k_function (int ∗xk, int i, int l, int k, llist ∗children, llist ∗solution_list)
- void count_base_k (int k, int l, llist ∗children, llist ∗solution_list)
- void count_base_kv (int ∗kv, int l, void function(int ∗x, int y, void ∗z, void ∗w), void ∗arg1, void ∗arg2)
- void markup_node_tree_recursive (tree ∗b, int ∗gcolors)
- void markup_node_tree (tree ∗b, int ∗gcolors, int vertices)
- void print_node_tree (tree ∗b, int ∗gcolors, int t)
- int get_combinations (llist ∗∗graphl, int vertices, int c, llist ∗combs)
- int hide_get_combinations (llist ∗∗graphl, int vertices, int c, llist ∗combs)
- int get_combinations_with (int n, llist ∗∗graphl, int vertices, int ∗gcolors, int c, llist ∗combs)
- void multipliers (int k, int d, int ∗sizes_d, int depth, tree ∗mults, llist ∗leafs)
- void n_choose_i (int n, int i, int l, tree ∗chose, llist ∗leafs)
- void mark_cmap (int c, llist ∗∗marked)
- void clear_cmarks (llist ∗marked)
- void dfs_on_marked (int n, llist ∗∗graphl)
- int check_connected_subgraph (llist ∗cgns, llist ∗∗graphl)
- int graph_connected (llist ∗∗graphl, int vertices)
- void get_distance_k_nodes_rec (llist ∗∗graphl, int node, int k, int vertices, llist ∗dirty, llist ∗collected, int dfo)

- llist ∗ get_distance_k_nodes (llist ∗∗graphl, int node, int k, int vertices)
- int get_bf_combinations_from_subgraphs (llist ∗∗graphl, int c, int vertices)
- int get_bf_combinations_list_8 (llist ∗∗graphl, llist ∗vertices)
- int get_bf_combinations_list_7 (llist ∗∗graphl, llist ∗vertices)
- int get_bf_combinations_list_6 (llist ∗∗graphl, llist ∗vertices)
- int get_bf_combinations_list_5 (llist ∗∗graphl, llist ∗vertices)
- int get_bf_combinations_list_4 (llist ∗∗graphl, llist ∗vertices)
- int get_bf_combinations_list_3 (llist ∗∗graphl, llist ∗vertices)
- int get_bf_combinations_list_2 (llist ∗∗graphl, llist ∗vertices)
- int get_bf_combinations_8 (llist ∗∗graphl, int vertices)
- int get_bf_combinations_7 (llist ∗∗graphl, int vertices)
- int get_bf_combinations_6 (llist ∗∗graphl, int vertices)
- int get_bf_combinations_5 (llist ∗∗graphl, int vertices)
- int get_bf_combinations_4 (llist ∗∗graphl, int vertices)
- int get_bf_combinations_3 (llist ∗∗graphl, int vertices)
- int get_bf_combinations_2 (llist ∗∗graphl, int vertices)

4.9.1 Function Documentation

4.9.1.1 int assign_next_join (int *x,* llist_node ∗∗ *assigned_values,* llist ∗∗ *domain_lists,* llist ∗∗ *assigned_current_domains*)

4.9.1.2 void backtrack_search_join (int *l,* llist ∗∗∗ *solutions,* llist ∗ *solutions_list*)

4.9.1.3 int backtrack_search_join_rec (int *l,* llist_node ∗∗ *assigned_values,* llist ∗∗ *assigned_current_domains,* llist ∗ *remaining_vars,* llist ∗∗∗ *solutions,* llist ∗ *solutions_list,* llist ∗∗ *domain_lists*)

4.9.1.4 int check_connected_subgraph (llist ∗ *cgns,* llist ∗∗ *graphl*)

4.9.1.5 int check_constraint (llist ∗ *combR,* llist ∗ *combL*)

4.9.1.6 int choose_var_join (llist ∗ *remaining_vars,* llist ∗∗ *domain_lists*)

4.9.1.7 void clear_cmarks (llist ∗ *marked*)

4.9.1.8 void collect_solutions (int ∗ *counts,* int *size,* llist ∗ *branches,* llist ∗ *solutions_list*)

4.9.1.9 void collect_solutions_with_all_combs (int ∗ *counts,* int *size,* llist ∗ *branches,* llist ∗ *solutions_list*)

4.9.1.10 int comb (llist ∗∗ *graphl,* int *c,* int *l,* int *k,* set ∗ *alpha,* tree ∗ *b*)

4.9.1.11 void combine_check_solution (int ∗ *counts,* int *size,* void ∗ *arg1,* void ∗ *arg2*)

4.9.1.12 void count_base_k (int *k,* int *l,* llist ∗ *children,* llist ∗ *solution_list*)

4.9.1.13 void count_base_k_function (int ∗ *xk,* int *i,* int *l,* int *k,* llist ∗ *children,* llist ∗ *solution_list*)

4.9.1.14 void count_base_kv (int ∗ *kv,* int *l,* void *functionint ∗x, int y, void ∗z, void ∗w,* void ∗ *arg1,* void ∗ *arg2*)

4.9.1.15 int dac_check_join (int *var,* int *l,* llist_node ∗∗ *assigned_values,* llist ∗∗ *domain_lists,* llist ∗∗ *undo_list,* int ∗ *cc,* llist ∗ *remaining_vars*)

4.9.1.16 void dfs_on_marked (int *n,* llist ∗∗ *graphl*)

4.9.1.17 int forward_check_join (int var, int l, llist_node ∗∗ assigned_values, llist ∗∗ domain_lists, llist ∗∗ undo_list, int ∗ cc, llist ∗ remaining_vars)

4.9.1.18 int get_bf_combinations_2 (llist ∗∗ graphl, int vertices)

4.9.1.19 int get_bf_combinations_3 (llist ∗∗ graphl, int vertices)

4.9.1.20 int get_bf_combinations_4 (llist ∗∗ graphl, int vertices)

4.9.1.21 int get_bf_combinations_5 (llist ∗∗ graphl, int vertices)

4.9.1.22 int get_bf_combinations_6 (llist ∗∗ graphl, int vertices)

4.9.1.23 int get_bf_combinations_7 (llist ∗∗ graphl, int vertices)

4.9.1.24 int get_bf_combinations_8 (llist ∗∗ graphl, int vertices)

4.9.1.25 int get_bf_combinations_from_subgraphs (llist ∗∗ graphl, int c, int vertices)

4.9.1.26 int get_bf_combinations_list_2 (llist ∗∗ graphl, llist ∗ vertices)

4.9.1.27 int get_bf_combinations_list_3 (llist ∗∗ graphl, llist ∗ vertices)

4.9.1.28 int get_bf_combinations_list_4 (llist ∗∗ graphl, llist ∗ vertices)

4.9.1.29 int get_bf_combinations_list_5 (llist ∗∗ graphl, llist ∗ vertices)

4.9.1.30 int get_bf_combinations_list_6 (llist ∗∗ graphl, llist ∗ vertices)

4.9.1.31 int get_bf_combinations_list_7 (llist ∗∗ graphl, llist ∗ vertices)

4.9.1.32 int get_bf_combinations_list_8 (llist ∗∗ graphl, llist ∗ vertices)

4.9.1.33 int get_combinations (llist ∗∗ graphl, int vertices, int c, llist ∗ combs)

4.9.1.34 int get_combinations_with (int n, llist ∗∗ graphl, int vertices, int ∗ gcolors, int c, llist ∗ combs)

4.9.1.35 llist∗ get_distance_k_nodes (llist ∗∗ graphl, int node, int k, int vertices)

4.9.1.36 void get_distance_k_nodes_rec (llist ∗∗ graphl, int node, int k, int vertices, llist ∗ dirty, llist ∗ collected, int dfo)

4.9.1.37 llist∗∗ get_solutions_of_size (tree ∗ n, int k)

4.9.1.38 int graph_connected (llist ∗∗ graphl, int vertices)

4.9.1.39 int hide_get_combinations (llist ∗∗ graphl, int vertices, int c, llist ∗ combs)

4.9.1.40 int mac_check_join (int var, int l, llist_node ∗∗ assigned_values, llist ∗∗ domain_lists, llist ∗∗ undo_list, int ∗ cc, llist ∗ remaining_vars)

4.9.1.41 void mark_cmap (int c, llist ∗∗ marked)

4.9.1.42 void markup_node_tree (tree ∗ b, int ∗ gcolors, int vertices)

4.9.1.43 void markup_node_tree_recursive (tree ∗ b, int ∗ gcolors)

4.9.1.44 void memo_count_base_k_function (int ∗ *xk,* int *i,* int *l,* int *k,* llist ∗ *children,* llist ∗ *solution_list*)

4.9.1.45 void multipliers (int *k,* int *d,* int ∗ *sizes_d,* int *depth,* tree ∗ *mults,* llist ∗ *leafs*)

4.9.1.46 void n_choose_i (int *n,* int *i,* int *l,* tree ∗ *chose,* llist ∗ *leafs*)

4.9.1.47 void print_combination (llist ∗ *comb*)

4.9.1.48 void print_combinations (llist ∗ *combs*)

4.9.1.49 void print_node_tree (tree ∗ *b,* int ∗ *gcolors,* int *t*)

4.9.1.50 void undo_filtering_join (int *l,* llist ∗∗ *undo_list,* llist ∗∗ *domain_lists*)

4.10 combinations.h File Reference

```
#include "llist.h"
#include "tree.h"
#include "set.h"
```

Classes

- struct solution_structure

Typedefs

- typedef struct solution_structure solution_structure

Functions

- int compare_nodes (void ∗va, void ∗vb)
- int comb (llist ∗∗graphl, int c, int l, int k, set ∗alpha, tree ∗b)
- void collect_solutions (int ∗counts, int size, llist ∗branches, llist ∗solutions_list)
- llist ∗∗ get_solutions_of_size (tree ∗n, int k)
- void combine_check_solution (int ∗counts, int size, void ∗arg1, void ∗arg2)
- void count_base_k (int k, int l, llist ∗children, llist ∗solution_list)
- void count_base_kv (int ∗kv, int l, void function(int ∗x, int y, void ∗z, void ∗w), void ∗arg1, void ∗arg2)
- int get_combinations (llist ∗∗graphl, int vertices, int c, llist ∗combs)
- int get_combinations_with (int n, llist ∗∗graphl, int vertices, int ∗gcolors, int c, llist ∗combs)
- void print_node_tree (tree ∗b, int ∗gcolors, int t)
- void markup_node_tree_recursive (tree ∗b, int ∗gcolors)
- void markup_node_tree (tree ∗b, int ∗gcolors, int vertices)
- void multipliers (int k, int d, int ∗sizes_d, int depth, tree ∗mults, llist ∗leafs)
- void collect_solutions_with_all_combs (int ∗counts, int size, llist ∗branches, llist ∗solutions_list)
- void count_base_k_function (int ∗xk, int i, int l, int k, llist ∗children, llist ∗solution_list)
- void memo_count_base_k_function (int ∗xk, int i, int l, int k, llist ∗children, llist ∗solution_list)
- void xxcount_base_k_function (int ∗xk, int i, int l, int k, void arg00(), void ∗arg01, void ∗arg02)
- void n_choose_i (int n, int i, int l, tree ∗chose, llist ∗leafs)
- int hide_get_combinations (llist ∗∗graphl, int vertices, int c, llist ∗combs)
- int get_bf_combinations_8 (llist ∗∗graphl, int vertices)
- int get_bf_combinations_7 (llist ∗∗graphl, int vertices)
- int get_bf_combinations_6 (llist ∗∗graphl, int vertices)

- int get_bf_combinations_5 (llist ∗∗graphl, int vertices)
- int get_bf_combinations_4 (llist ∗∗graphl, int vertices)
- int get_bf_combinations_3 (llist ∗∗graphl, int vertices)
- int get_bf_combinations_2 (llist ∗∗graphl, int vertices)
- void mark_cmap (int c, llist ∗∗marked)
- void clear_cmarks (llist ∗marked)
- void dfs_on_marked (int n, llist ∗∗graphl)
- int check_connected_subgraph (llist ∗cgns, llist ∗∗graphl)
- int graph_connected (llist ∗∗graphl, int vertices)
- int sac_check_join (int var, int l, llist_node ∗∗assigned_values, llist ∗∗domain_lists, llist ∗∗undo_list, int ∗cc)
- int dac_check_join (int var, int l, llist_node ∗∗assigned_values, llist ∗∗domain_lists, llist ∗∗undo_list, int ∗cc, llist ∗remaining_vars)
- int forward_check_join (int var, int l, llist_node ∗∗assigned_values, llist ∗∗domain_lists, llist ∗∗undo_list, int ∗cc, llist ∗remaining_vars)
- int assign_next_join (int x, llist_node ∗∗assigned_values, llist ∗∗domain_lists, llist ∗∗assigned_current_domains)
- void undo_filtering_join (int l, llist ∗∗undo_list, llist ∗∗domain_lists)
- int fc_join (int l, llist ∗∗∗solutions, llist ∗solutions_list, llist ∗∗passed_domain_lists)
- int mac_check_join (int var, int l, llist_node ∗∗assigned_values, llist ∗∗domain_lists, llist ∗∗undo_list, int ∗cc, llist ∗remaining_vars)
- int check_constraint (llist ∗comba, llist ∗combb)
- int revise_combs (int ∗tuple, llist ∗∗∗solutions, int ∗∗∗last, char ∗∗alive)
- int apply_ac_join (llist ∗∗∗solutions, set ∗queue)
- void print_combinations (llist ∗combs)
- void print_combination (llist ∗comb)
- int choose_var_join (llist ∗remaining_vars, llist ∗∗domain_lists)
- void get_distance_k_nodes_rec (llist ∗∗graphl, int node, int k, int vertices, llist ∗dirty, llist ∗collected, int dfo)
- llist ∗ get_distance_k_nodes (llist ∗∗graphl, int node, int k, int vertices)
- int get_bf_combinations_from_subgraphs (llist ∗∗graphl, int c, int vertices)
- int get_bf_combinations_list_8 (llist ∗∗graphl, llist ∗vertices)
- int get_bf_combinations_list_7 (llist ∗∗graphl, llist ∗vertices)
- int get_bf_combinations_list_6 (llist ∗∗graphl, llist ∗vertices)
- int get_bf_combinations_list_5 (llist ∗∗graphl, llist ∗vertices)
- int get_bf_combinations_list_4 (llist ∗∗graphl, llist ∗vertices)
- int get_bf_combinations_list_3 (llist ∗∗graphl, llist ∗vertices)
- int get_bf_combinations_list_2 (llist ∗∗graphl, llist ∗vertices)
- void backtrack_search_join (int l, llist ∗∗∗solutions, llist ∗solutions_list)
- int backtrack_search_join_rec (int l, llist_node ∗∗assigned_values, llist ∗∗assigned_current_domains, llist ∗remaining_vars, llist ∗∗∗solutions, llist ∗solutions_list, llist ∗∗domain_lists)

4.10.1 Detailed Description

Provides the implementations for several algorithms for computing connected combinations of nodes.

4.10.2 Typedef Documentation

4.10.2.1 typedef struct **solution_structure solution_structure**

4.10.3 Function Documentation

4.10.3.1 int apply_ac_join (**llist** ∗∗∗ *solutions,* **set** ∗ *queue*)

4.10.3.2 int assign_next_join (int *x,* llist_node ∗∗ *assigned_values,* llist ∗∗ *domain_lists,* llist ∗∗ *assigned_current_domains*)

4.10.3.3 void backtrack_search_join (int *l,* llist ∗∗∗ *solutions,* llist ∗ *solutions_list*)

4.10.3.4 int backtrack_search_join_rec (int *l,* llist_node ∗∗ *assigned_values,* llist ∗∗ *assigned_current_domains,* llist ∗ *remaining_vars,* llist ∗∗∗ *solutions,* llist ∗ *solutions_list,* llist ∗∗ *domain_lists*)

4.10.3.5 int check_connected_subgraph (llist ∗ *cgns,* llist ∗∗ *graphl*)

4.10.3.6 int check_constraint (llist ∗ *comba,* llist ∗ *combb*)

4.10.3.7 int choose_var_join (llist ∗ *remaining_vars,* llist ∗∗ *domain_lists*)

4.10.3.8 void clear_cmarks (llist ∗ *marked*)

4.10.3.9 void collect_solutions (int ∗ *counts,* int *size,* llist ∗ *branches,* llist ∗ *solutions_list*)

4.10.3.10 void collect_solutions_with_all_combs (int ∗ *counts,* int *size,* llist ∗ *branches,* llist ∗ *solutions_list*)

4.10.3.11 int comb (llist ∗∗ *graphl,* int *c,* int *l,* int *k,* set ∗ *alpha,* tree ∗ *b*)

4.10.3.12 void combine_check_solution (int ∗ *counts,* int *size,* void ∗ *arg1,* void ∗ *arg2*)

4.10.3.13 int compare_nodes (void ∗ *va,* void ∗ *vb*)

4.10.3.14 void count_base_k (int *k,* int *l,* llist ∗ *children,* llist ∗ *solution_list*)

4.10.3.15 void count_base_k_function (int ∗ *xk,* int *i,* int *l,* int *k,* llist ∗ *children,* llist ∗ *solution_list*)

4.10.3.16 void count_base_kv (int ∗ *kv,* int *l,* void *functionint ∗x, int y, void ∗z, void ∗w,* void ∗ *arg1,* void ∗ *arg2*)

4.10.3.17 int dac_check_join (int *var,* int *l,* llist_node ∗∗ *assigned_values,* llist ∗∗ *domain_lists,* llist ∗∗ *undo_list,* int ∗ *cc,* llist ∗ *remaining_vars*)

4.10.3.18 void dfs_on_marked (int *n,* llist ∗∗ *graphl*)

4.10.3.19 int fc_join (int *l,* llist ∗∗∗ *solutions,* llist ∗ *solutions_list,* llist ∗∗ *passed_domain_lists*)

4.10.3.20 int forward_check_join (int *var,* int *l,* llist_node ∗∗ *assigned_values,* llist ∗∗ *domain_lists,* llist ∗∗ *undo_list,* int ∗ *cc,* llist ∗ *remaining_vars*)

4.10.3.21 int get_bf_combinations_2 (llist ∗∗ *graphl,* int *vertices*)

4.10.3.22 int get_bf_combinations_3 (llist ∗∗ *graphl,* int *vertices*)

4.10.3.23 int get_bf_combinations_4 (llist ∗∗ *graphl,* int *vertices*)

4.10.3.24 int get_bf_combinations_5 (llist ∗∗ *graphl,* int *vertices*)

4.10.3.25 int get_bf_combinations_6 (llist ∗∗ *graphl,* int *vertices*)

4.10.3.26 int get_bf_combinations_7 (llist ∗∗ *graphl,* int *vertices*)

4.10.3.27 int get_bf_combinations_8 (llist ∗∗ *graphl,* int *vertices*)

4.10.3.28 int get_bf_combinations_from_subgraphs (llist ** graphl, int c, int vertices)

4.10.3.29 int get_bf_combinations_list_2 (llist ** graphl, llist * vertices)

4.10.3.30 int get_bf_combinations_list_3 (llist ** graphl, llist * vertices)

4.10.3.31 int get_bf_combinations_list_4 (llist ** graphl, llist * vertices)

4.10.3.32 int get_bf_combinations_list_5 (llist ** graphl, llist * vertices)

4.10.3.33 int get_bf_combinations_list_6 (llist ** graphl, llist * vertices)

4.10.3.34 int get_bf_combinations_list_7 (llist ** graphl, llist * vertices)

4.10.3.35 int get_bf_combinations_list_8 (llist ** graphl, llist * vertices)

4.10.3.36 int get_combinations (llist ** graphl, int vertices, int c, llist * combs)

4.10.3.37 int get_combinations_with (int n, llist ** graphl, int vertices, int * gcolors, int c, llist * combs)

4.10.3.38 llist* get_distance_k_nodes (llist ** graphl, int node, int k, int vertices)

4.10.3.39 void get_distance_k_nodes_rec (llist ** graphl, int node, int k, int vertices, llist * dirty, llist * collected, int dfo)

4.10.3.40 llist** get_solutions_of_size (tree * n, int k)

4.10.3.41 int graph_connected (llist ** graphl, int vertices)

4.10.3.42 int hide_get_combinations (llist ** graphl, int vertices, int c, llist * combs)

4.10.3.43 int mac_check_join (int var, int l, llist_node ** assigned_values, llist ** domain_lists, llist ** undo_list, int * cc, llist * remaining_vars)

4.10.3.44 void mark_cmap (int c, llist ** marked)

4.10.3.45 void markup_node_tree (tree * b, int * gcolors, int vertices)

4.10.3.46 void markup_node_tree_recursive (tree * b, int * gcolors)

4.10.3.47 void memo_count_base_k_function (int * xk, int i, int l, int k, llist * children, llist * solution_list)

4.10.3.48 void multipliers (int k, int d, int * sizes_d, int depth, tree * mults, llist * leafs)

4.10.3.49 void n_choose_i (int n, int i, int l, tree * chose, llist * leafs)

4.10.3.50 void print_combination (llist * comb)

4.10.3.51 void print_combinations (llist * combs)

4.10.3.52 void print_node_tree (tree * b, int * gcolors, int t)

4.10.3.53 int revise_combs (int * tuple, llist *** solutions, int *** last, char ** alive)

4.10.3.54 int sac_check_join (int var, int l, llist_node ** assigned_values, llist ** domain_lists, llist ** undo_list, int * cc)

4.10.3.55 void undo_filtering_join (int *l*, **llist** ∗∗ *undo_list*, **llist** ∗∗ *domain_lists*)

4.10.3.56 void xxcount_base_k_function (int ∗ *xk*, int *i*, int *l*, int *k*, void *arg00()*, void ∗ *arg01*, void ∗ *arg02*)

4.11 constraints.c File Reference

```
#include "constraints.h"
#include "variables.h"
#include "utils.h"
#include "query_graph.h"
#include "decomposition.h"
```

Functions

- constraint ∗ new_constraint (char ∗name, int id, int ∗vars, int fill)
- constraints ∗ init_constraints (int count)
- constraint ∗ add_constraint (char ∗name, int function, constraints ∗consts, int arity, int const_id, int parameter_count, tuple ∗tuples, int ac3)
- void add_variable_to_last_constraint (constraints ∗consts, variables ∗vars, int var_id, int ac3)
- void add_parameter_to_last_constraint (constraints ∗consts, void ∗value, int value_type, int pos)
- void reset_ac3_1_last (constraints ∗consts, variables ∗vars)
- llist ∗ split_constraints_into_connected_components (constraints ∗consts, int set_var_constraints)
- llist ∗ split_constraints_into_connected_components_list (llist ∗consts, int set_var_constraints)
- void split_constraints_into_connected_components_rec (constraint ∗cons, char ∗map, llist ∗maplist)
- void add_htables_to_constraints (void)
- void printClusterGraphNeato (llist ∗constraints)
- node_graph ∗ dual_node_graph_from_constraints (llist ∗constraints)
- constraint_tree ∗ get_constraint_tree (llist ∗constraints)
- constraint_tree ∗ new_constraint_tree ()
- constraint_tree_node ∗ new_constraint_tree_node (llist ∗clique)
- constraint_tree ∗ get_dual_join_tree (llist ∗cliques)
- void print_consistant_tree (constraint_tree ∗ct)
- void constraints_make_this_root (constraint_tree_node ∗newroot)
- void constraints_swap_parent_child (constraint_tree_node ∗child)
- int get_unassinged_htables (constraint_tree_node ∗node)
- void print_number_of_deleted_tuples_values (llist ∗usedhts_list)
- void add_constraint_to_variable (variable ∗var, constraint ∗constr)

Variables

- int constraint_ids = 0

4.11.1 Function Documentation

4.11.1.1 **constraint**∗ add_constraint (char ∗ *name*, int *function*, **constraints** ∗ *consts*, int *arity*, int *const_id*, int *parameter_count*, tuple ∗ *tuples*, int *ac3*)

4.11.1.2 void add_constraint_to_variable (**variable** ∗ *var*, **constraint** ∗ *constr*)

4.11.1.3 void add_htables_to_constraints (void)

4.11.1.4 void add_parameter_to_last_constraint (**constraints** ∗ *consts*, void ∗ *value*, int *value_type*, int *pos*)

4.11.1.5 void add_variable_to_last_constraint (**constraints** ∗ *consts,* **variables** ∗ *vars,* int *var_id,* int *ac3*)

4.11.1.6 void constraints_make_this_root (**constraint_tree_node** ∗ *newroot*)

4.11.1.7 void constraints_swap_parent_child (**constraint_tree_node** ∗ *child*)

4.11.1.8 **node_graph**∗ dual_node_graph_from_constraints (**llist** ∗ *constraints*) `[read]`

4.11.1.9 **constraint_tree**∗ get_constraint_tree (**llist** ∗ *constraints*)

4.11.1.10 **constraint_tree**∗ get_dual_join_tree (**llist** ∗ *cliques*)

4.11.1.11 int get_unassinged_htables (**constraint_tree_node** ∗ *node*)

4.11.1.12 **constraints**∗ init_constraints (int *count*)

4.11.1.13 **constraint**∗ new_constraint (char ∗ *name,* int *id,* int ∗ *vars,* int *fill*)

4.11.1.14 **constraint_tree**∗ new_constraint_tree (void)

4.11.1.15 **constraint_tree_node**∗ new_constraint_tree_node (**llist** ∗ *clique*)

4.11.1.16 void print_consistant_tree (**constraint_tree** ∗ *ct*)

4.11.1.17 void print_number_of_deleted_tuples_values (**llist** ∗ *usedhts_list*)

4.11.1.18 void printClusterGraphNeato (**llist** ∗ *constraints*)

4.11.1.19 void reset_ac3_1_last (**constraints** ∗ *consts,* **variables** ∗ *vars*)

4.11.1.20 **llist**∗ split_constraints_into_connected_components (**constraints** ∗ *consts,* int *set_var_constraints*)

4.11.1.21 **llist**∗ split_constraints_into_connected_components_list (**llist** ∗ *consts,* int *set_var_constraints*)

4.11.1.22 void split_constraints_into_connected_components_rec (**constraint** ∗ *cons,* char ∗ *map,* **llist** ∗ *maplist*)

4.11.2 Variable Documentation

4.11.2.1 int constraint_ids = 0

4.12 constraints.h File Reference

```
#include <stdlib.h>
#include "variables.h"
#include "relations.h"
```

Classes

- struct parameter
- struct constraint
- struct constraints
- struct constraint_tree
- struct constraint_tree_node

Typedefs

- typedef struct parameter parameter
- typedef struct constraint constraint
- typedef struct constraints constraints
- typedef struct constraint_tree constraint_tree
- typedef struct constraint_tree_node constraint_tree_node

Functions

- constraint ∗ new_constraint (char ∗name, int id, int ∗vars, int fill)
- constraints ∗ init_constraints (int count)
- constraint ∗ add_constraint (char ∗name, int function, constraints ∗consts, int arity, int const_id, int parameter_count, tuple ∗tuples, int ac3)
- void add_variable_to_last_constraint (constraints ∗consts, variables ∗vars, int variable, int ac3)
- void add_parameter_to_last_constraint (constraints ∗consts, void ∗value, int type, int pos)
- void reset_ac3_1_last (constraints ∗consts, variables ∗vars)
- llist ∗ split_constraints_into_connected_components (constraints ∗consts, int set_variable_hashtable)
- llist ∗ split_constraints_into_connected_components_list (llist ∗consts, int set_variable_hashtable)
- void split_constraints_into_connected_components_rec (constraint ∗cons, char ∗map, llist ∗maplist)
- void add_htables_to_constraints (void)
- void printClusterGraphNeato (llist ∗constraints)
- struct node_graph ∗ dual_node_graph_from_constraints (llist ∗constraints)
- constraint_tree ∗ get_constraint_tree (llist ∗constraints)
- constraint_tree ∗ get_dual_join_tree (llist ∗cliques)
- constraint_tree ∗ new_constraint_tree (void)
- constraint_tree_node ∗ new_constraint_tree_node (llist ∗clique)
- void print_consistant_tree (constraint_tree ∗ct)
- void constraints_swap_parent_child (constraint_tree_node ∗child)
- void constraints_make_this_root (constraint_tree_node ∗newroot)
- int get_unassinged_htables (constraint_tree_node ∗node)
- void print_number_of_deleted_tuples_values (llist ∗usedhts_list)
- void add_constraint_to_variable (variable ∗var, constraint ∗constraint)

4.12.1 Detailed Description

Provides the data structures for representing the CSP constraints and methods for constructing and manipulating them.

4.12.2 Typedef Documentation

4.12.2.1 typedef struct constraint constraint

4.12.2.2 typedef struct constraint_tree constraint_tree

4.12.2.3 typedef struct constraint_tree_node constraint_tree_node

4.12.2.4 typedef struct constraints constraints

4.12.2.5 typedef struct parameter parameter

4.12.3 Function Documentation

4.12.3.1 constraint∗ add_constraint (char ∗ *name*, int *function*, constraints ∗ *consts*, int *arity*, int *const_id*, int *parameter_count*, tuple ∗ *tuples*, int *ac3*)

4.12.3.2 void add_constraint_to_variable (variable ∗ *var*, constraint ∗ *constraint*)

4.12.3.3 void add_htables_to_constraints (void)

4.12.3.4 void add_parameter_to_last_constraint (constraints ∗ *consts*, void ∗ *value*, int *type*, int *pos*)

4.12.3.5 void add_variable_to_last_constraint (constraints ∗ *consts*, variables ∗ *vars*, int *variable*, int *ac3*)

4.12.3.6 void constraints_make_this_root (constraint_tree_node ∗ *newroot*)

4.12.3.7 void constraints_swap_parent_child (constraint_tree_node ∗ *child*)

4.12.3.8 struct node_graph∗ dual_node_graph_from_constraints (llist ∗ *constraints*) [read]

4.12.3.9 constraint_tree∗ get_constraint_tree (llist ∗ *constraints*)

4.12.3.10 constraint_tree∗ get_dual_join_tree (llist ∗ *cliques*)

4.12.3.11 int get_unassinged_htables (constraint_tree_node ∗ *node*)

4.12.3.12 constraints∗ init_constraints (int *count*)

4.12.3.13 constraint∗ new_constraint (char ∗ *name*, int *id*, int ∗ *vars*, int *fill*)

4.12.3.14 constraint_tree∗ new_constraint_tree (void)

4.12.3.15 constraint_tree_node∗ new_constraint_tree_node (llist ∗ *clique*)

4.12.3.16 void print_consistant_tree (constraint_tree ∗ *ct*)

4.12.3.17 void print_number_of_deleted_tuples_values (llist ∗ *usedhts_list*)

4.12.3.18 void printClusterGraphNeato (llist ∗ *constraints*)

4.12.3.19 void reset_ac3_1_last (constraints ∗ *consts*, variables ∗ *vars*)

4.12.3.20 llist∗ split_constraints_into_connected_components (constraints ∗ *consts*, int *set_variable_hashtable*)

4.12.3.21 llist∗ split_constraints_into_connected_components_list (llist ∗ *consts*, int *set_variable_hashtable*)

4.12.3.22 void split_constraints_into_connected_components_rec (constraint ∗ *cons*, char ∗ *map*, llist ∗ *maplist*)

4.13 decomposition.c File Reference

```
#include "decomposition.h"
#include "query_graph.h"
#include "utils.h"
#include "combinations_wrapper.h"
#include "rs2.h"
#include "tree_decomp.h"
#include "rsprocedures.h"
#include "tree.h"
#include "active_tuple.h"
#include "splitter.h"
#include "minibucket.h"
#include "partition.h"
#include "decomposition_functions.h"
#include "hashtable_search.h"
#include "tables.h"
#include "sort.h"
```

Functions

- void set_dual_heap_position (int pos, void ∗body)
- void set_dual_heap_value (int val, void ∗body)
- int get_dual_heap_value (void ∗body)
- constraint_graph_node ∗∗ get_dual_elimination_ordering (constraint_graph ∗cg)
- int count_dual_fill_edges (char ∗∗matrix, constraint_graph_node ∗node)
- void add_dual_fill_edges (char ∗∗matrix, constraint_graph_node ∗node, heap ∗h, constraint_graph ∗cg)
- llist ∗ dual_cliques (constraint_graph_node ∗∗ordering, constraint_graph ∗cg)
- node_graph ∗ new_node_graph (constraint_graph ∗cg)
- node_graph ∗ new_node_graph_wpeo (constraint_graph ∗cg, node_graph ∗old_graph_get_node_peo_from, llist ∗added_edges)
- node_graph ∗ new_node_graph_from_rels (llist ∗rels, llist ∗vars, int ∗varmap)
- void add_fill_edges_to_node_graph (node_graph ∗ng, llist ∗filledges)
- void destroy_node_graph (node_graph ∗ng)
- void set_heap_position (int pos, void ∗body)
- int get_heap_value (void ∗body)
- void print_node_graph (node_graph ∗ng)
- int ∗ get_elimination_ordering (node_graph ∗ng, llist ∗added_edges)
- int count_fill_edges (node_graph ∗ng, int var)
- void add_fill_edges (node_graph ∗ng, heap ∗h, int var, llist ∗added_edges)
- void ng_set_heap_pos (int pos, void ∗body)
- void ng_set_heap_value (int val, void ∗body)
- int ng_get_heap_value (void ∗body)
- int ∗ max_cardinality (node_graph ∗ng, int node)
- llist ∗ cliques (int ∗ordering, node_graph ∗ng, int ∗size)
- int get_common_rels_count_from_sets (set ∗va, set ∗vb, llist ∗clist)
- int get_common_vars_count_from_sets (set ∗va, set ∗vb, llist ∗clist)
- void add_decomposed_tree_child (decomposed_tree ∗child, decomposed_tree ∗parent)
- decomposed_tree ∗ new_decomposed_tree (llist ∗vl)
- void set_leafs (decomposed_tree ∗root)
- decomposed_tree ∗∗ get_cluster_array (decomposed_tree ∗root)
- void make_graph (decomposed_tree ∗root, llist ∗leafs, decomposed_tree ∗∗∗c_array)
- void make_combinations_for_cluster (decomposed_tree ∗cluster, llist ∗combs)
- int compare_combs (void ∗o1, void ∗o2)

- int get_comb_node_id (void ∗body, void ∗bla)
- int setup_combinations (decomposed_tree ∗∗c_array, int clusters)
- void add_separator_relations (decomposed_tree ∗∗c_array, int clusters)
- void add_separator_relations_for_cluster (decomposed_tree ∗current, int clusters, int clear_combdatas)
- decomposed_tree ∗ build_decomposed_tree_vertices (llist ∗cliques, llist ∗tree_nodes)
- decomposed_tree ∗ find_smallest_root (llist ∗nodes)
- decomposed_tree ∗ find_balancing_root (llist ∗nodes)
- decomposed_tree ∗ find_max_degree_node (llist ∗nodes)
- void make_this_root (decomposed_tree ∗newroot)
- void swap_parent_child (decomposed_tree ∗child, decomposed_tree ∗parent)
- void merge_tree_nodes (decomposed_tree ∗root)
- void reid_decomposed_tree_nodes (decomposed_tree ∗root, int ∗id)
- void merge_tree_nodes_rec (decomposed_tree ∗treen)
- void merge_with_parent (decomposed_tree ∗treen)
- void print_decomposed_tree_node (decomposed_tree ∗tree, int tab, decomposition_info ∗di)
- void print_decomposed_tree_node_dot (decomposed_tree ∗tree, decomposition_info ∗di, FILE ∗f)
- void print_decomposed_tree_node_summary_dot (decomposed_tree ∗tree, decomposition_info ∗di, FILE ∗f)
- void print_decomposed_tree_cluster_dot (decomposed_tree ∗tree)
- void print_decomposed_tree_cluster (decomposed_tree ∗tree)
- decomposition_info ∗ new_di (void)
- void print_decomposed_tree (decomposed_tree ∗tree, int tab, FILE ∗f, int summary, int print_screen)
- void print_decomposed_tree_rec (decomposed_tree ∗tree, int tab, int summary, decomposition_info ∗di, FILE ∗f)
- void set_uncovered_variables (decomposed_tree ∗root)
- void report_separator_quality (decomposed_tree ∗child)
- void print_filtering_report (constraints ∗cons)
- void check_all_constraints_added (constraints ∗cons, FILE ∗f)
- decomposed_tree ∗ get_decomposed_tree_from_library (constraint_graph ∗cg, llist ∗tree_nodes, int tree_only_no_cg)
- decomposed_tree ∗ get_decomposed_tree_from_library_rec (tree_node ∗tree, constraint_graph ∗cg, llist ∗tree_nodes, int tree_only_no_cg)
- decomposed_tree ∗ get_dual_decomposed_tree_from_library (constraint_graph ∗cg, llist ∗tree_nodes, int tree_only_no_cg)
- decomposed_tree ∗ get_dual_decomposed_tree_from_library_rec (tree_node ∗tree, constraint_graph ∗cg, llist ∗tree_nodes, int tree_only_no_cg)
- int label_levels (decomposed_tree ∗root)
- int add_affiliate_rels (llist ∗tree_nodes)
- void append_to_comb_head (htable ∗ht, llist ∗comb)
- void append_to_comb (htable ∗ht, llist ∗comb)
- void hide_comb_datas (decomposed_tree ∗node)
- void un_hide_comb_datas (decomposed_tree ∗node)
- void cluster_pre_process (decomposed_tree ∗treen, int copy_tables)
- void cluster_pre_process2 (decomposed_tree ∗treen, int copy_tables)
- int set_subtree_size (decomposed_tree ∗root)
- void label_depth (decomposed_tree ∗root)
- void clear_combdatas_from_relations (decomposed_tree ∗root)
- void clear_combdatas_from_relations_only_root (decomposed_tree ∗root, int onlyroot)
- void split_trees (decomposed_tree ∗tree, llist ∗trees)
- void print_decomposition_tree_tables_h (decomposed_tree ∗treen)
- void print_decomposition_tree_tables (decomposed_tree ∗treen)
- void normalized_projected_rels_on_c_vars (decomposed_tree ∗treen)
- void set_pre_post (decomposed_tree ∗treen, int ∗id, decomposed_tree ∗∗treens)
- int in_subtree (decomposed_tree ∗root, decomposed_tree ∗node)
- void generate_xml (decomposed_tree ∗tree, int idp, int idp2, int idp3)
- void report_tree_separators (decomposed_tree ∗treen)
- void report_tree_separators2 (decomposed_tree ∗treen)
- void print_tree_report (decomposed_tree ∗tree, double ∗maxratio, int ∗maxcv, int ∗maxsv)

4.13.1 Function Documentation

4.13.1.1 int add_affiliate_rels (llist ∗ *tree_nodes*)

4.13.1.2 void add_decomposed_tree_child (decomposed_tree ∗ *child,* decomposed_tree ∗ *parent*)

4.13.1.3 void add_dual_fill_edges (char ∗∗ *matrix,* constraint_graph_node ∗ *node,* heap ∗ *h,* constraint_graph ∗ *cg*)

4.13.1.4 void add_fill_edges (node_graph ∗ *ng,* heap ∗ *h,* int *var,* llist ∗ *added_edges*)

4.13.1.5 void add_fill_edges_to_node_graph (node_graph ∗ *ng,* llist ∗ *filledges*)

4.13.1.6 void add_separator_relations (decomposed_tree ∗∗ *c_array,* int *clusters*)

4.13.1.7 void add_separator_relations_for_cluster (decomposed_tree ∗ *current,* int *clusters,* int *clear_combdatas*)

4.13.1.8 void append_to_comb (htable ∗ *ht,* llist ∗ *comb*)

4.13.1.9 void append_to_comb_head (htable ∗ *ht,* llist ∗ *comb*)

4.13.1.10 decomposed_tree∗ build_decomposed_tree_vertices (llist ∗ *cliques,* llist ∗ *tree_nodes*)

4.13.1.11 void check_all_constraints_added (constraints ∗ *cons,* FILE ∗ *f*)

4.13.1.12 void clear_combdatas_from_relations (decomposed_tree ∗ *root*)

4.13.1.13 void clear_combdatas_from_relations_only_root (decomposed_tree ∗ *root,* int *onlyroot*)

4.13.1.14 llist∗ cliques (int ∗ *ordering,* node_graph ∗ *ng,* int ∗ *size*)

4.13.1.15 void cluster_pre_process (decomposed_tree ∗ *treen,* int *copy_tables*)

4.13.1.16 void cluster_pre_process2 (decomposed_tree ∗ *treen,* int *copy_tables*)

4.13.1.17 int compare_combs (void ∗ *o1,* void ∗ *o2*)

4.13.1.18 int count_dual_fill_edges (char ∗∗ *matrix,* constraint_graph_node ∗ *node*)

4.13.1.19 int count_fill_edges (node_graph ∗ *ng,* int *var*)

4.13.1.20 void destroy_node_graph (node_graph ∗ *ng*)

4.13.1.21 llist∗ dual_cliques (constraint_graph_node ∗∗ *ordering,* constraint_graph ∗ *cg*)

4.13.1.22 decomposed_tree∗ find_balancing_root (llist ∗ *nodes*)

4.13.1.23 decomposed_tree∗ find_max_degree_node (llist ∗ *nodes*)

4.13.1.24 decomposed_tree∗ find_smallest_root (llist ∗ *nodes*)

4.13.1.25 void generate_xml (decomposed_tree ∗ *tree,* int *idp,* int *idp2,* int *idp3*)

4.13.1.26 decomposed_tree∗∗ get_cluster_array (decomposed_tree ∗ *root*)

4.13.1.27 int get_comb_node_id (void ∗ *body,* void ∗ *bla*)

4.13.1.28 int get_common_rels_count_from_sets (set ∗ *va,* set ∗ *vb,* llist ∗ *clist*)

4.13.1.29 int get_common_vars_count_from_sets (set ∗ *va,* set ∗ *vb,* llist ∗ *clist*)

4.13.1.30 decomposed_tree∗ get_decomposed_tree_from_library (constraint_graph ∗ *cg,* llist ∗ *tree_nodes,* int *tree_only_no_cg*)

4.13.1.31 decomposed_tree∗ get_decomposed_tree_from_library_rec (tree_node ∗ *tree,* constraint_graph ∗ *cg,* llist ∗ *tree_nodes,* int *tree_only_no_cg*)

4.13.1.32 decomposed_tree∗ get_dual_decomposed_tree_from_library (constraint_graph ∗ *cg,* llist ∗ *tree_nodes,* int *tree_only_no_cg*)

4.13.1.33 decomposed_tree∗ get_dual_decomposed_tree_from_library_rec (tree_node ∗ *tree,* constraint_graph ∗ *cg,* llist ∗ *tree_nodes,* int *tree_only_no_cg*)

4.13.1.34 constraint_graph_node∗∗ get_dual_elimination_ordering (constraint_graph ∗ *cg*)

4.13.1.35 int get_dual_heap_value (void ∗ *body*)

4.13.1.36 int∗ get_elimination_ordering (node_graph ∗ *ng,* llist ∗ *added_edges*)

4.13.1.37 int get_heap_value (void ∗ *body*)

4.13.1.38 void hide_comb_datas (decomposed_tree ∗ *node*)

4.13.1.39 int in_subtree (decomposed_tree ∗ *root,* decomposed_tree ∗ *node*)

4.13.1.40 void label_depth (decomposed_tree ∗ *root*)

4.13.1.41 int label_levels (decomposed_tree ∗ *root*)

4.13.1.42 void make_combinations_for_cluster (decomposed_tree ∗ *cluster,* llist ∗ *combs*)

4.13.1.43 void make_graph (decomposed_tree ∗ *root,* llist ∗ *leafs,* decomposed_tree ∗∗∗ *c_array*)

4.13.1.44 void make_this_root (decomposed_tree ∗ *newroot*)

4.13.1.45 int∗ max_cardinality (node_graph ∗ *ng,* int *node*)

4.13.1.46 void merge_tree_nodes (decomposed_tree ∗ *root*)

4.13.1.47 void merge_tree_nodes_rec (decomposed_tree ∗ *treen*)

4.13.1.48 void merge_with_parent (decomposed_tree ∗ *treen*)

4.13.1.49 decomposed_tree∗ new_decomposed_tree (llist ∗ *vl*)

4.13.1.50 decomposition_info∗ new_di (void)

4.13.1.51 node_graph∗ new_node_graph (constraint_graph ∗ *cg*)

4.13.1.52 node_graph∗ new_node_graph_from_rels (llist ∗ *rels,* llist ∗ *vars,* int ∗ *varmap*)

4.13.1.53 node_graph∗ new_node_graph_wpeo (constraint_graph ∗ *cg,* node_graph ∗ *old_graph_get_node_peo_from,* llist ∗ *added_edges*)

4.13.1.54 int ng_get_heap_value (void ∗ *body*)

4.13.1.55 void ng_set_heap_pos (int *pos,* void ∗ *body*)

4.13.1.56 void ng_set_heap_value (int *val,* void ∗ *body*)

4.13.1.57 void normalized_projected_rels_on_c_vars (decomposed_tree ∗ *treen*)

4.13.1.58 void print_decomposed_tree (decomposed_tree ∗ *tree,* int *tab,* FILE ∗ *f,* int *summary,* int *print_screen*)

4.13.1.59 void print_decomposed_tree_cluster (decomposed_tree ∗ *tree*)

4.13.1.60 void print_decomposed_tree_cluster_dot (decomposed_tree ∗ *tree*)

4.13.1.61 void print_decomposed_tree_node (decomposed_tree ∗ *tree,* int *tab,* decomposition_info ∗ *di*)

4.13.1.62 void print_decomposed_tree_node_dot (decomposed_tree ∗ *tree,* decomposition_info ∗ *di,* FILE ∗ *f*)

4.13.1.63 void print_decomposed_tree_node_summary_dot (decomposed_tree ∗ *tree,* decomposition_info ∗ *di,* FILE ∗ *f*)

4.13.1.64 void print_decomposed_tree_rec (decomposed_tree ∗ *tree,* int *tab,* int *summary,* decomposition_info ∗ *di,* FILE ∗ *f*)

4.13.1.65 void print_decomposition_tree_tables (decomposed_tree ∗ *treen*)

4.13.1.66 void print_decomposition_tree_tables_h (decomposed_tree ∗ *treen*)

4.13.1.67 void print_filtering_report (constraints ∗ *cons*)

4.13.1.68 void print_node_graph (node_graph ∗ *ng*)

4.13.1.69 void print_tree_report (decomposed_tree ∗ *tree,* double ∗ *maxratio,* int ∗ *maxcv,* int ∗ *maxsv*)

4.13.1.70 void reid_decomposed_tree_nodes (decomposed_tree ∗ *root,* int ∗ *id*)

4.13.1.71 void report_separator_quality (decomposed_tree ∗ *child*)

4.13.1.72 void report_tree_separators (decomposed_tree ∗ *treen*)

4.13.1.73 void report_tree_separators2 (decomposed_tree ∗ *treen*)

4.13.1.74 void set_dual_heap_position (int *pos,* void ∗ *body*)

4.13.1.75 void set_dual_heap_value (int *val,* void ∗ *body*)

4.13.1.76 void set_heap_position (int *pos,* void ∗ *body*)

4.13.1.77 void set_leafs (decomposed_tree ∗ *root*)

4.13.1.78 void set_pre_post (decomposed_tree ∗ *treen,* int ∗ *id,* decomposed_tree ∗∗ *treens*)

4.13.1.79 int set_subtree_size (decomposed_tree ∗ *root*)

4.13.1.80 void set_uncovered_variables (decomposed_tree ∗ *root*)

4.13.1.81 int setup_combinations (**decomposed_tree** ∗∗ *c_array,* int *clusters*)

4.13.1.82 void split_trees (**decomposed_tree** ∗ *tree,* **llist** ∗ *trees*)

4.13.1.83 void swap_parent_child (**decomposed_tree** ∗ *child,* **decomposed_tree** ∗ *parent*)

4.13.1.84 void un_hide_comb_datas (**decomposed_tree** ∗ *node*)

4.14 decomposition.h File Reference

```
#include "llist.h"
#include "query_graph.h"
#include "heap.h"
#include "set.h"
#include "tree_decomp.h"
#include "rs.h"
#include "rb_tree.h"
```

Classes

- struct decomposition_info
- struct node_graph
- struct separator_info
- struct decomposed_tree

Typedefs

- typedef struct decomposition_info decomposition_info
- typedef struct node_graph node_graph
- typedef struct separator_info separator_info
- typedef struct decomposed_tree decomposed_tree

Functions

- constraint_graph_node ∗∗ get_dual_elimination_ordering (constraint_graph ∗cg)
- int count_dual_fill_edges (char ∗∗matrix, constraint_graph_node ∗node)
- void set_dual_heap_position (int pos, void ∗body)
- void set_dual_heap_value (int val, void ∗body)
- int get_dual_heap_value (void ∗body)
- void add_dual_fill_edges (char ∗∗matrix, constraint_graph_node ∗node, heap ∗h, constraint_graph ∗cg)
- llist ∗ dual_cliques (constraint_graph_node ∗∗ordering, constraint_graph ∗cg)
- void print_node_graph (node_graph ∗ng)
- node_graph ∗ new_node_graph (constraint_graph ∗cg)
- node_graph ∗ new_node_graph_wpeo (constraint_graph ∗cg, node_graph ∗old_graph_get_node_peo_from, llist ∗added_edges)
- void destroy_node_graph (node_graph ∗ng)
- int ∗ get_elimination_ordering (node_graph ∗ng, llist ∗added_edges)
- int count_fill_edges (node_graph ∗ng, int var)
- void set_heap_position (int pos, void ∗body)
- int get_heap_value (void ∗body)
- void add_fill_edges (node_graph ∗ng, heap ∗h, int var, llist ∗added_edges)
- llist ∗ cliques (int ∗ordering, node_graph ∗ng, int ∗size)

- int get_common_vars_count_from_sets (set ∗va, set ∗vb, llist ∗clist)
- int get_common_rels_count_from_sets (set ∗va, set ∗vb, llist ∗clist)
- void add_decomposed_tree_child (decomposed_tree ∗child, decomposed_tree ∗parent)
- void print_decomposed_tree_node_dot (decomposed_tree ∗tree, decomposition_info ∗di, FILE ∗f)
- void print_decomposed_tree_node_summary_dot (decomposed_tree ∗tree, decomposition_info ∗di, FILE ∗f)
- void print_decomposed_tree (decomposed_tree ∗tree, int tab, FILE ∗f, int summary, int print_screen)
- void print_decomposed_tree_rec (decomposed_tree ∗tree, int tab, int summary, decomposition_info ∗di, FILE ∗f)
- decomposed_tree ∗ build_decomposed_tree_vertices (llist ∗cliques, llist ∗tree_nodes)
- decomposed_tree ∗ new_decomposed_tree (llist ∗vl)
- void print_decomposed_tree_node (decomposed_tree ∗tree, int tab, decomposition_info ∗di)
- decomposition_info ∗ new_di (void)
- void print_filtering_report (constraints ∗cons)
- void check_all_constraints_added (constraints ∗cons, FILE ∗f)
- decomposed_tree ∗ find_max_degree_node (llist ∗nodes)
- void make_this_root (decomposed_tree ∗newroot)
- void swap_parent_child (decomposed_tree ∗child, decomposed_tree ∗parent)
- decomposed_tree ∗ get_decomposed_tree_from_library (constraint_graph ∗cg, llist ∗tree_nodes, int tree_only_no_cg)
- decomposed_tree ∗ get_decomposed_tree_from_library_rec (tree_node ∗tree, constraint_graph ∗cg, llist ∗tree_nodes, int tree_only_no_cg)
- void set_uncovered_variables (decomposed_tree ∗root)
- decomposed_tree ∗ find_balancing_root (llist ∗nodes)
- void init_cluster_queue (decomposed_tree ∗tree)
- int label_levels (decomposed_tree ∗root)
- int add_affiliate_rels (llist ∗tree_nodes)
- void append_to_comb (htable ∗ht, llist ∗comb)
- void hide_comb_datas (decomposed_tree ∗node)
- void un_hide_comb_datas (decomposed_tree ∗node)
- int filter_tuples_for_var_val (variable ∗var, value ∗val)
- void ng_set_heap_pos (int pos, void ∗body)
- void ng_set_heap_value (int val, void ∗body)
- int ng_get_heap_value (void ∗body)
- int ∗ max_cardinality (node_graph ∗ng, int node)
- void append_to_comb_head (htable ∗ht, llist ∗comb)
- void print_decomposed_tree_cluster_dot (decomposed_tree ∗tree)
- void print_decomposed_tree_cluster (decomposed_tree ∗tree)
- decomposed_tree ∗ get_dual_decomposed_tree_from_library (constraint_graph ∗cg, llist ∗tree_nodes, int tree_only_no_cg)
- decomposed_tree ∗ get_dual_decomposed_tree_from_library_rec (tree_node ∗tree, constraint_graph ∗cg, llist ∗tree_nodes, int tree_only_no_cg)
- void cluster_pre_process (decomposed_tree ∗treen, int copy_tables)
- void report_separator_quality (decomposed_tree ∗child)
- node_graph ∗ new_node_graph_from_rels (llist ∗rels, llist ∗vars, int ∗varmap)
- void add_fill_edges_to_node_graph (node_graph ∗ng, llist ∗filledges)
- void label_depth (decomposed_tree ∗root)
- void clear_combdatas_from_relations (decomposed_tree ∗root)
- void clear_combdatas_from_relations_only_root (decomposed_tree ∗root, int onlyroot)
- decomposed_tree ∗ find_smallest_root (llist ∗nodes)
- void split_trees (decomposed_tree ∗tree, llist ∗trees)
- void print_decomposition_tree_tables_h (decomposed_tree ∗treen)
- void print_decomposition_tree_tables (decomposed_tree ∗treen)
- int set_subtree_size (decomposed_tree ∗root)
- void normalized_projected_rels_on_c_vars (decomposed_tree ∗treen)
- void cluster_pre_process2 (decomposed_tree ∗treen, int copy_tables)

- void set_pre_post (decomposed_tree ∗treen, int ∗id, decomposed_tree ∗∗treens)
- int in_subtree (decomposed_tree ∗root, decomposed_tree ∗node)
- void make_graph (decomposed_tree ∗root, llist ∗leafs, decomposed_tree ∗∗∗array)
- void set_leafs (decomposed_tree ∗root)
- decomposed_tree ∗∗ get_cluster_array (decomposed_tree ∗root)
- void add_separator_relations (decomposed_tree ∗∗c_array, int clusters)
- void add_separator_relations_for_cluster (decomposed_tree ∗current, int clusters, int clear_combdatas)
- int setup_combinations (decomposed_tree ∗∗c_array, int clusters)
- void generate_xml (decomposed_tree ∗tree, int idp, int idp2, int idp3)
- void make_combinations_for_cluster (decomposed_tree ∗cluster, llist ∗combs)
- void report_tree_separators (decomposed_tree ∗treen)
- void report_tree_separators2 (decomposed_tree ∗treen)
- int get_comb_node_id (void ∗body, void ∗bla)
- void merge_tree_nodes (decomposed_tree ∗root)
- void reid_decomposed_tree_nodes (decomposed_tree ∗root, int ∗id)
- void merge_tree_nodes_rec (decomposed_tree ∗root)
- void merge_with_parent (decomposed_tree ∗treen)
- void print_tree_report (decomposed_tree ∗tree, double ∗maxratio, int ∗maxcv, int ∗maxsv)

4.14.1 Detailed Description

Provides the implementation for the tree clustering data-structure and the algorithms for building them.

4.14.2 Typedef Documentation

4.14.2.1 typedef struct **decomposed_tree decomposed_tree**

4.14.2.2 typedef struct **decomposition_info decomposition_info**

4.14.2.3 typedef struct **node_graph node_graph**

4.14.2.4 typedef struct **separator_info separator_info**

4.14.3 Function Documentation

4.14.3.1 int add_affiliate_rels (**llist** ∗ *tree_nodes*)

4.14.3.2 void add_decomposed_tree_child (**decomposed_tree** ∗ *child,* **decomposed_tree** ∗ *parent*)

4.14.3.3 void add_dual_fill_edges (char ∗∗ *matrix,* **constraint_graph_node** ∗ *node,* **heap** ∗ *h,* **constraint_graph** ∗ *cg*)

4.14.3.4 void add_fill_edges (**node_graph** ∗ *ng,* **heap** ∗ *h,* int *var,* **llist** ∗ *added_edges*)

4.14.3.5 void add_fill_edges_to_node_graph (**node_graph** ∗ *ng,* **llist** ∗ *filledges*)

4.14.3.6 void add_separator_relations (**decomposed_tree** ∗∗ *c_array,* int *clusters*)

4.14.3.7 void add_separator_relations_for_cluster (**decomposed_tree** ∗ *current,* int *clusters,* int *clear_combdatas*)

4.14.3.8 void append_to_comb (**htable** ∗ *ht,* **llist** ∗ *comb*)

4.14.3.9 void append_to_comb_head (**htable** ∗ *ht,* **llist** ∗ *comb*)

4.14.3.10 decomposed_tree* build_decomposed_tree_vertices (llist * cliques, llist * tree_nodes)

4.14.3.11 void check_all_constraints_added (constraints * cons, FILE * f)

4.14.3.12 void clear_combdatas_from_relations (decomposed_tree * root)

4.14.3.13 void clear_combdatas_from_relations_only_root (decomposed_tree * root, int onlyroot)

4.14.3.14 llist* cliques (int * ordering, node_graph * ng, int * size)

4.14.3.15 void cluster_pre_process (decomposed_tree * treen, int copy_tables)

4.14.3.16 void cluster_pre_process2 (decomposed_tree * treen, int copy_tables)

4.14.3.17 int count_dual_fill_edges (char ** matrix, constraint_graph_node * node)

4.14.3.18 int count_fill_edges (node_graph * ng, int var)

4.14.3.19 void destroy_node_graph (node_graph * ng)

4.14.3.20 llist* dual_cliques (constraint_graph_node ** ordering, constraint_graph * cg)

4.14.3.21 int filter_tuples_for_var_val (variable * var, value * val)

4.14.3.22 decomposed_tree* find_balancing_root (llist * nodes)

4.14.3.23 decomposed_tree* find_max_degree_node (llist * nodes)

4.14.3.24 decomposed_tree* find_smallest_root (llist * nodes)

4.14.3.25 void generate_xml (decomposed_tree * tree, int idp, int idp2, int idp3)

4.14.3.26 decomposed_tree** get_cluster_array (decomposed_tree * root)

4.14.3.27 int get_comb_node_id (void * body, void * bla)

4.14.3.28 int get_common_rels_count_from_sets (set * va, set * vb, llist * clist)

4.14.3.29 int get_common_vars_count_from_sets (set * va, set * vb, llist * clist)

4.14.3.30 decomposed_tree* get_decomposed_tree_from_library (constraint_graph * cg, llist * tree_nodes, int tree_only_no_cg)

4.14.3.31 decomposed_tree* get_decomposed_tree_from_library_rec (tree_node * tree, constraint_graph * cg, llist * tree_nodes, int tree_only_no_cg)

4.14.3.32 decomposed_tree* get_dual_decomposed_tree_from_library (constraint_graph * cg, llist * tree_nodes, int tree_only_no_cg)

4.14.3.33 decomposed_tree* get_dual_decomposed_tree_from_library_rec (tree_node * tree, constraint_graph * cg, llist * tree_nodes, int tree_only_no_cg)

4.14.3.34 constraint_graph_node** get_dual_elimination_ordering (constraint_graph * cg)

4.14.3.35 int get_dual_heap_value (void * body)

4.14.3.36 int* get_elimination_ordering (node_graph * *ng,* llist * *added_edges*)

4.14.3.37 int get_heap_value (void * *body*)

4.14.3.38 void hide_comb_datas (decomposed_tree * *node*)

4.14.3.39 int in_subtree (decomposed_tree * *root,* decomposed_tree * *node*)

4.14.3.40 void init_cluster_queue (decomposed_tree * *tree*)

4.14.3.41 void label_depth (decomposed_tree * *root*)

4.14.3.42 int label_levels (decomposed_tree * *root*)

4.14.3.43 void make_combinations_for_cluster (decomposed_tree * *cluster,* llist * *combs*)

4.14.3.44 void make_graph (decomposed_tree * *root,* llist * *leafs,* decomposed_tree *** *array*)

4.14.3.45 void make_this_root (decomposed_tree * *newroot*)

4.14.3.46 int* max_cardinality (node_graph * *ng,* int *node*)

4.14.3.47 void merge_tree_nodes (decomposed_tree * *root*)

4.14.3.48 void merge_tree_nodes_rec (decomposed_tree * *root*)

4.14.3.49 void merge_with_parent (decomposed_tree * *treen*)

4.14.3.50 decomposed_tree* new_decomposed_tree (llist * *vl*)

4.14.3.51 decomposition_info* new_di (void)

4.14.3.52 node_graph* new_node_graph (constraint_graph * *cg*)

4.14.3.53 node_graph* new_node_graph_from_rels (llist * *rels,* llist * *vars,* int * *varmap*)

4.14.3.54 node_graph* new_node_graph_wpeo (constraint_graph * *cg,* node_graph * *old_graph_get_node_peo_from,* llist * *added_edges*)

4.14.3.55 int ng_get_heap_value (void * *body*)

4.14.3.56 void ng_set_heap_pos (int *pos,* void * *body*)

4.14.3.57 void ng_set_heap_value (int *val,* void * *body*)

4.14.3.58 void normalized_projected_rels_on_c_vars (decomposed_tree * *treen*)

4.14.3.59 void print_decomposed_tree (decomposed_tree * *tree,* int *tab,* FILE * *f,* int *summary,* int *print_screen*)

4.14.3.60 void print_decomposed_tree_cluster (decomposed_tree * *tree*)

4.14.3.61 void print_decomposed_tree_cluster_dot (decomposed_tree * *tree*)

4.14.3.62 void print_decomposed_tree_node (decomposed_tree * *tree,* int *tab,* decomposition_info * *di*)

4.14.3.63 void print_decomposed_tree_node_dot (decomposed_tree * *tree,* decomposition_info * *di,* FILE * *f*)

4.14.3.64 void print_decomposed_tree_node_summary_dot (**decomposed_tree** ∗ *tree,* **decomposition_info** ∗ *di,* FILE ∗ *f*)

4.14.3.65 void print_decomposed_tree_rec (**decomposed_tree** ∗ *tree,* int *tab,* int *summary,* **decomposition_info** ∗ *di,* FILE ∗ *f*)

4.14.3.66 void print_decomposition_tree_tables (**decomposed_tree** ∗ *treen*)

4.14.3.67 void print_decomposition_tree_tables_h (**decomposed_tree** ∗ *treen*)

4.14.3.68 void print_filtering_report (**constraints** ∗ *cons*)

4.14.3.69 void print_node_graph (**node_graph** ∗ *ng*)

4.14.3.70 void print_tree_report (**decomposed_tree** ∗ *tree,* double ∗ *maxratio,* int ∗ *maxcv,* int ∗ *maxsv*)

4.14.3.71 void reid_decomposed_tree_nodes (**decomposed_tree** ∗ *root,* int ∗ *id*)

4.14.3.72 void report_separator_quality (**decomposed_tree** ∗ *child*)

4.14.3.73 void report_tree_separators (**decomposed_tree** ∗ *treen*)

4.14.3.74 void report_tree_separators2 (**decomposed_tree** ∗ *treen*)

4.14.3.75 void set_dual_heap_position (int *pos,* void ∗ *body*)

4.14.3.76 void set_dual_heap_value (int *val,* void ∗ *body*)

4.14.3.77 void set_heap_position (int *pos,* void ∗ *body*)

4.14.3.78 void set_leafs (**decomposed_tree** ∗ *root*)

4.14.3.79 void set_pre_post (**decomposed_tree** ∗ *treen,* int ∗ *id,* **decomposed_tree** ∗∗ *treens*)

4.14.3.80 int set_subtree_size (**decomposed_tree** ∗ *root*)

4.14.3.81 void set_uncovered_variables (**decomposed_tree** ∗ *root*)

4.14.3.82 int setup_combinations (**decomposed_tree** ∗∗ *c_array,* int *clusters*)

4.14.3.83 void split_trees (**decomposed_tree** ∗ *tree,* **llist** ∗ *trees*)

4.14.3.84 void swap_parent_child (**decomposed_tree** ∗ *child,* **decomposed_tree** ∗ *parent*)

4.14.3.85 void un_hide_comb_datas (**decomposed_tree** ∗ *node*)

4.15 decomposition_functions.c File Reference

```
#include "decomposition_functions.h"
#include "decomposition.h"
#include "query_graph.h"
#include "utils.h"
#include "combinations_wrapper.h"
#include "rs2.h"
#include "tree_decomp.h"
#include "rsprocedures.h"
#include "tree.h"
#include "active_tuple.h"
#include "splitter.h"
#include "minibucket.h"
#include "partition.h"
#include "hashtable_search.h"
#include <math.h>
#include "rs2tree.h"
#include "tables.h"
```

Functions

- void add_constraints_to_cluique_pD_old_not_used (decomposed_tree ∗clique, constraint_graph ∗cg)
- void add_constraints_to_cluique_pE (decomposed_tree ∗clique, constraint_graph ∗cg, int filter_from_omitted)
- void add_constraints_to_cluique_pB (decomposed_tree ∗clique)
- void add_constraints_to_cluique_pA (decomposed_tree ∗clique, constraint_graph ∗cg)
- void add_constraints_to_cluique_pC (decomposed_tree ∗clique, constraint_graph ∗cg)
- void add_constraints_to_cluiques (decomposed_tree ∗tree, llist ∗added_edges, constraint_graph ∗cg)
- int process_cluster_generate_separator (decomposed_tree ∗treen)
- int process_cluster_filter_separators (decomposed_tree ∗treen)
- void generate_solutions (decomposed_tree ∗treen, active_tuple ∗ac)
- void generate_solutions2 (decomposed_tree ∗treen, active_tuple ∗ac)
- int combinations_splitter (decomposed_tree ∗treen)
- void set_all_relation_scope_separator (decomposed_tree ∗treen)
- void isolate_relations (decomposed_tree ∗treen)

4.15.1 Function Documentation

4.15.1.1 void add_constraints_to_cluique_pA (**decomposed_tree** ∗ *clique,* **constraint_graph** ∗ *cg*)

4.15.1.2 void add_constraints_to_cluique_pB (**decomposed_tree** ∗ *clique*)

4.15.1.3 void add_constraints_to_cluique_pC (**decomposed_tree** ∗ *clique,* **constraint_graph** ∗ *cg*)

4.15.1.4 void add_constraints_to_cluique_pD_old_not_used (**decomposed_tree** ∗ *clique,* **constraint_graph** ∗ *cg*)

4.15.1.5 void add_constraints_to_cluique_pE (**decomposed_tree** ∗ *clique,* **constraint_graph** ∗ *cg,* int *filter_from_omitted*)

4.15.1.6 void add_constraints_to_cluiques (**decomposed_tree** ∗ *tree,* **llist** ∗ *added_edges,* **constraint_graph** ∗ *cg*)

4.15.1.7 int combinations_splitter (**decomposed_tree** ∗ *treen*)

4.15.1.8 void generate_solutions (decomposed_tree ∗ *treen,* active_tuple ∗ *ac*)

4.15.1.9 void generate_solutions2 (decomposed_tree ∗ *treen,* active_tuple ∗ *ac*)

4.15.1.10 void isolate_relations (decomposed_tree ∗ *treen*)

4.15.1.11 int process_cluster_filter_separators (decomposed_tree ∗ *treen*)

4.15.1.12 int process_cluster_generate_separator (decomposed_tree ∗ *treen*)

4.15.1.13 void set_all_relation_scope_separator (decomposed_tree ∗ *treen*)

4.16 decomposition_functions.h File Reference

```
#include "utils.h"
#include "decomposition.h"
```

Functions

- void add_constraints_to_cluique_pA (decomposed_tree ∗clique, constraint_graph ∗cg)
- void add_constraints_to_cluique_pD_old_not_used (decomposed_tree ∗clique, constraint_graph ∗cg)
- void add_constraints_to_cluique_pD (decomposed_tree ∗clique, constraint_graph ∗cg)
- void add_constraints_to_cluique_pB (decomposed_tree ∗clique)
- void add_constraints_to_cluique_pC (decomposed_tree ∗clique, constraint_graph ∗cg)
- void add_constraints_to_cluique_pE (decomposed_tree ∗clique, constraint_graph ∗cg, int filter_from_omitted)
- void add_constraints_to_cliuques (decomposed_tree ∗tree, llist ∗added_edges, constraint_graph ∗cg)
- int process_cluster_generate_separator (decomposed_tree ∗treen)
- int process_cluster_filter_separators (decomposed_tree ∗treen)
- void generate_solutions (decomposed_tree ∗treen, active_tuple ∗ac)
- void generate_solutions2 (decomposed_tree ∗treen, active_tuple ∗ac)
- int combinations_splitter (decomposed_tree ∗treen)
- void set_all_relation_scope_separator (decomposed_tree ∗treen)
- void isolate_relations (decomposed_tree ∗tree)

4.16.1 Detailed Description

Implements helper methods used with the tree clustering.

4.16.2 Function Documentation

4.16.2.1 void add_constraints_to_cluique_pA (decomposed_tree ∗ *clique,* constraint_graph ∗ *cg*)

4.16.2.2 void add_constraints_to_cluique_pB (decomposed_tree ∗ *clique*)

4.16.2.3 void add_constraints_to_cluique_pC (decomposed_tree ∗ *clique,* constraint_graph ∗ *cg*)

4.16.2.4 void add_constraints_to_cluique_pD (decomposed_tree ∗ *clique,* constraint_graph ∗ *cg*)

4.16.2.5 void add_constraints_to_cluique_pD_old_not_used (decomposed_tree ∗ *clique,* constraint_graph ∗ *cg*)

4.16.2.6 void add_constraints_to_clique_pE (**decomposed_tree** ∗ *clique,* **constraint_graph** ∗ *cg,* int *filter_from_omitted*)

4.16.2.7 void add_constraints_to_cliques (**decomposed_tree** ∗ *tree,* **llist** ∗ *added_edges,* **constraint_graph** ∗ *cg*)

4.16.2.8 int combinations_splitter (**decomposed_tree** ∗ *treen*)

4.16.2.9 void generate_solutions (**decomposed_tree** ∗ *treen,* **active_tuple** ∗ *ac*)

4.16.2.10 void generate_solutions2 (**decomposed_tree** ∗ *treen,* **active_tuple** ∗ *ac*)

4.16.2.11 void isolate_relations (**decomposed_tree** ∗ *tree*)

4.16.2.12 int process_cluster_filter_separators (**decomposed_tree** ∗ *treen*)

4.16.2.13 int process_cluster_generate_separator (**decomposed_tree** ∗ *treen*)

4.16.2.14 void set_all_relation_scope_separator (**decomposed_tree** ∗ *treen*)

4.17 do_db.c File Reference

```
#include "do_db.h"
#include "utils.h"
#include "query_graph.h"
#include "rpwc.h"
#include "rs.h"
#include "rsprocedures.h"
#include "rpwcprocedures.h"
#include "combinations_wrapper.h"
#include "rs2.h"
#include "tree.h"
#include "tree_decomp.h"
#include "db_algo1.h"
#include "decomposition.h"
#include "btdallsol.h"
#include "splitter.h"
#include "decomposition_functions.h"
#include "rs2tree.h"
#include "septables.h"
#include "gbtd.h"
```

Functions

- int do_db (main_structure ∗m_s)
- int do_tree (main_structure ∗m_s)

4.17.1 Function Documentation

4.17.1.1 int do_db (**main_structure** ∗ *m_s*)

4.17.1.2 int do_tree (**main_structure** ∗ *m_s*)

4.18 do_db.h File Reference

```
#include "globals.h"
```

Functions

- int do_db (main_structure ∗m_s)
- int do_tree (main_structure ∗m_s)

4.18.1 Detailed Description

Provides the interfaces for initializing the tree-decomposition.

4.18.2 Function Documentation

4.18.2.1 int do_db (**main_structure** ∗ ***m_s***)

4.18.2.2 int do_tree (**main_structure** ∗ ***m_s***)

4.19 domains.c File Reference

```
#include "domains.h"
#include "utils.h"
```

Functions

- llist ∗ new_domains (void)
- void add_domain_name (llist ∗domains, char ∗name)
- void add_domain_int_list_to_last_domain (llist ∗domains, int from, int to)
- value ∗ value_copy (value ∗val)

4.19.1 Function Documentation

4.19.1.1 void add_domain_int_list_to_last_domain (**llist** ∗ ***domains,*** int ***from,*** int ***to***)

4.19.1.2 void add_domain_name (**llist** ∗ ***domains,*** char ∗ ***name***)

4.19.1.3 **llist**∗ new_domains (void)

4.19.1.4 **value**∗ value_copy (**value** ∗ ***val***)

4.20 domains.h File Reference

```
#include "llist.h"
```

Classes

- struct value

Typedefs

- typedef struct value value

Functions

- llist ∗ new_domains (void)
- void add_domain_name (llist ∗domains, char ∗name)
- void add_domain_int_list_to_last_domain (llist ∗domains, int from, int to)
- value ∗ value_copy (value ∗val)

4.20.1 Detailed Description

Provides the data structures for representing and domains of variables.

4.20.2 Typedef Documentation

4.20.2.1 typedef struct **value value**

4.20.3 Function Documentation

4.20.3.1 void add_domain_int_list_to_last_domain (**llist** ∗ *domains,* int *from,* int *to*)

4.20.3.2 void add_domain_name (**llist** ∗ *domains,* char ∗ *name*)

4.20.3.3 **llist**∗ new_domains (void)

4.20.3.4 **value**∗ value_copy (**value** ∗ *val*)

4.21 gbtd.c File Reference

```
#include "gbtd.h"
#include "utils.h"
#include "variables.h"
#include "rs2tree.h"
#include "rs2.h"
#include "rpwcprocedures.h"
#include "btdallsol.h"
```

Functions

- gbtd_system ∗ new_gbtd_system (void)
- pseudo_cluster ∗ new_pseudo_cluster (void)
- void destroy_pseudo_cluster (pseudo_cluster ∗ps)
- gbtd_system ∗ init_gbtd (llist ∗vars, int allSol, options system, set ∗queue, set ∗∗queues, int total_clusters, decomposed_tree ∗∗treens, comb_data ∗cd, int revision_time, int ∗tuple_positions, int ∗tuple_position_htid, active_tuple ∗ac, int count_on_m_s)
- void destroy_gbtd_system (gbtd_system ∗gs)
- void generate_pseudo_tree (gbtd_system ∗gs)
- int run_gbtd (gbtd_system ∗gs)
- int run_gbtd_wtn (gbtd_system ∗gs)
- int current_has_next (gbtd_system ∗gs)

- int filter_gvars (gbtd_system ∗gs, int skip_if_all_revised)
- pseudo_cluster ∗ sibling (pseudo_cluster ∗pc)
- void find_gvar (gbtd_system ∗gs)
- void find_gvar_wtn (gbtd_system ∗gs, int backwards)
- void advance_current_domain (gbtd_system ∗gs)
- void destroy_var_undo_data (var_undo_data ∗vud)
- void move_to_previous_var (gbtd_system ∗gs)
- void reset_gvar (gbtd_system ∗gs, void ∗gvar)
- void assign_gvar (gbtd_system ∗gs)
- htv_undo_data ∗ new_htv_undo_data (htvar ∗htv, gbtd_system ∗gs)
- var_undo_data ∗ new_var_undo_data (variable ∗var)
- int unassign_subtree (gbtd_system ∗gs)
- void print_gbtd_system (gbtd_system ∗gs)
- void undo_htv_assignment (htv_undo_data ∗vud, gbtd_system ∗gs)
- htvar ∗ new_htvar (htable ∗ht, int id, llist_node ∗chosen_value)
- void destroy_htvar (htvar ∗htv)
- void do_something_with_solution (gbtd_system ∗gs)
- void init_goods_nogoods (pseudo_cluster ∗psc)
- void set_good_nogood (pseudo_cluster ∗psc)
- int has_nogood (pseudo_cluster ∗psc)
- double has_good (pseudo_cluster ∗psc)
- int relation_var_tuple_stamp (gbtd_system ∗gs, htvar ∗htv, int ∗tuple, int first, int marked)
- int covers_sep_relations (htvar ∗htv, gbtd_system ∗gs)
- int no_unassigned_cover_variable (gbtd_system ∗gs)
- void count_goods_nogoods (gbtd_system ∗gs)
- void count_goods_nogoods_rec (pseudo_cluster ∗root, double ∗countnogood, double ∗countgood)

4.21.1 Function Documentation

4.21.1.1 void advance_current_domain (**gbtd_system** ∗ *gs*)

4.21.1.2 void assign_gvar (**gbtd_system** ∗ *gs*)

4.21.1.3 void count_goods_nogoods (**gbtd_system** ∗ *gs*)

4.21.1.4 void count_goods_nogoods_rec (**pseudo_cluster** ∗ *root,* double ∗ *countnogood,* double ∗ *countgood*)

4.21.1.5 int covers_sep_relations (**htvar** ∗ *htv,* **gbtd_system** ∗ *gs*)

4.21.1.6 int current_has_next (**gbtd_system** ∗ *gs*)

4.21.1.7 void destroy_gbtd_system (**gbtd_system** ∗ *gs*)

4.21.1.8 void destroy_htvar (**htvar** ∗ *htv*)

4.21.1.9 void destroy_pseudo_cluster (**pseudo_cluster** ∗ *ps*)

4.21.1.10 void destroy_var_undo_data (**var_undo_data** ∗ *vud*)

4.21.1.11 void do_something_with_solution (**gbtd_system** ∗ *gs*)

4.21.1.12 int filter_gvars (**gbtd_system** ∗ *gs,* int *skip_if_all_revised*)

4.21.1.13 void find_gvar (**gbtd_system** ∗ *gs*)

4.21.1.14 void find_gvar_wtn (gbtd_system ∗ *gs*, int *backwards*)

4.21.1.15 void generate_pseudo_tree (gbtd_system ∗ *gs*)

4.21.1.16 double has_good (pseudo_cluster ∗ *psc*)

4.21.1.17 int has_nogood (pseudo_cluster ∗ *psc*)

4.21.1.18 gbtd_system∗ init_gbtd (llist ∗ *vars*, int *allSol*, options *system*, set ∗ *queue*, set ∗∗ *queues*, int *total_clusters*, decomposed_tree ∗∗ *treens*, comb_data ∗ *cd*, int *revision_time*, int ∗ *tuple_positions*, int ∗ *tuple_position_htid*, active_tuple ∗ *ac*, int *count_on_m_s*)

4.21.1.19 void init_goods_nogoods (pseudo_cluster ∗ *psc*)

4.21.1.20 void move_to_previous_var (gbtd_system ∗ *gs*)

4.21.1.21 gbtd_system∗ new_gbtd_system (void)

4.21.1.22 htv_undo_data∗ new_htv_undo_data (htvar ∗ *htv*, gbtd_system ∗ *gs*)

4.21.1.23 htvar∗ new_htvar (htable ∗ *ht*, int *id*, llist_node ∗ *chosen_value*)

4.21.1.24 pseudo_cluster∗ new_pseudo_cluster (void)

4.21.1.25 var_undo_data∗ new_var_undo_data (variable ∗ *var*)

4.21.1.26 int no_unassigned_cover_variable (gbtd_system ∗ *gs*)

4.21.1.27 void print_gbtd_system (gbtd_system ∗ *gs*)

4.21.1.28 int relation_var_tuple_stamp (gbtd_system ∗ *gs*, htvar ∗ *htv*, int ∗ *tuple*, int *first*, int *marked*)

4.21.1.29 void reset_gvar (gbtd_system ∗ *gs*, void ∗ *gvar*)

4.21.1.30 int run_gbtd (gbtd_system ∗ *gs*)

4.21.1.31 int run_gbtd_wtn (gbtd_system ∗ *gs*)

4.21.1.32 void set_good_nogood (pseudo_cluster ∗ *psc*)

4.21.1.33 pseudo_cluster∗ sibling (pseudo_cluster ∗ *pc*)

4.21.1.34 int unassign_subtree (gbtd_system ∗ *gs*)

4.21.1.35 void undo_htv_assignment (htv_undo_data ∗ *vud*, gbtd_system ∗ *gs*)

4.22 gbtd.h File Reference

```
#include "rb_tree.h"
#include "llist.h"
#include "decomposition.h"
```

Classes

- struct pseudo_cluster
- struct gbtd_system
- struct var_undo_data
- struct htvar
- struct htv_undo_data

Typedefs

- typedef struct pseudo_cluster pseudo_cluster
- typedef struct gbtd_system gbtd_system
- typedef struct var_undo_data var_undo_data
- typedef struct htvar htvar
- typedef struct htv_undo_data htv_undo_data

Functions

- gbtd_system ∗ init_gbtd (llist ∗vars, int allSol, options system, set ∗queue, set ∗∗queues, int total_clusters, decomposed_tree ∗∗treens, comb_data ∗cd, int revision_time, int ∗tuple_positions, int ∗tuple_position_htid, active_tuple ∗ac, int count_on_m_s)
- void move_to_previous_var (gbtd_system ∗gs)
- pseudo_cluster ∗ sibling (pseudo_cluster ∗pc)
- int unassign_subtree (gbtd_system ∗gs)
- void assign_gvar (gbtd_system ∗gs)
- int filter_gvars (gbtd_system ∗gs, int skip_if_all_revised)
- void advance_current_domain (gbtd_system ∗gs)
- void reset_gvar (gbtd_system ∗gs, void ∗gvar)
- void find_gvar (gbtd_system ∗gs)
- void find_gvar_wtn (gbtd_system ∗gs, int backwards)
- gbtd_system ∗ new_gbtd_system (void)
- pseudo_cluster ∗ new_pseudo_cluster (void)
- void generate_pseudo_tree (gbtd_system ∗gs)
- int run_gbtd (gbtd_system ∗gs)
- int run_gbtd_wtn (gbtd_system ∗gs)
- int current_has_next (gbtd_system ∗gs)
- void destroy_var_undo_data (var_undo_data ∗vud)
- var_undo_data ∗ new_var_undo_data (variable ∗var)
- void print_gbtd_system (gbtd_system ∗gs)
- void destroy_pseudo_cluster (pseudo_cluster ∗ps)
- void destroy_gbtd_system (gbtd_system ∗gs)
- void undo_htv_assignment (htv_undo_data ∗vud, gbtd_system ∗gs)
- htv_undo_data ∗ new_htv_undo_data (htvar ∗htv, gbtd_system ∗gs)
- htvar ∗ new_htvar (htable ∗ht, int id, llist_node ∗chosen_value)
- void destroy_htvar (htvar ∗htv)
- void do_something_with_solution (gbtd_system ∗gs)
- double has_good (pseudo_cluster ∗psc)
- int has_nogood (pseudo_cluster ∗psc)
- void set_good_nogood (pseudo_cluster ∗psc)
- void init_goods_nogoods (pseudo_cluster ∗psc)
- int relation_var_tuple_stamp (gbtd_system ∗gs, htvar ∗htv, int ∗tuple, int first, int marked)
- int no_unassigned_cover_variable (gbtd_system ∗gs)
- int covers_sep_relations (htvar ∗htv, gbtd_system ∗gs)
- void count_goods_nogoods_rec (pseudo_cluster ∗root, double ∗countnogood, double ∗countgood)
- void count_goods_nogoods (gbtd_system ∗gs)

4.22.1 Detailed Description

Provides a generalized backtracking with tree decomposition algorithm. Used for most backtrack-based algorithms with and without tree-decomposition both for variables and relations. Also implements the core of PerTuple and AllSol algorithms.

4.22.2 Typedef Documentation

4.22.2.1 typedef struct gbtd_system gbtd_system

4.22.2.2 typedef struct htv_undo_data htv_undo_data

4.22.2.3 typedef struct htvar htvar

4.22.2.4 typedef struct pseudo_cluster pseudo_cluster

4.22.2.5 typedef struct var_undo_data var_undo_data

4.22.3 Function Documentation

4.22.3.1 void advance_current_domain (gbtd_system ∗ *gs*)

4.22.3.2 void assign_gvar (gbtd_system ∗ *gs*)

4.22.3.3 void count_goods_nogoods (gbtd_system ∗ *gs*)

4.22.3.4 void count_goods_nogoods_rec (pseudo_cluster ∗ *root,* double ∗ *countnogood,* double ∗ *countgood*)

4.22.3.5 int covers_sep_relations (htvar ∗ *htv,* gbtd_system ∗ *gs*)

4.22.3.6 int current_has_next (gbtd_system ∗ *gs*)

4.22.3.7 void destroy_gbtd_system (gbtd_system ∗ *gs*)

4.22.3.8 void destroy_htvar (htvar ∗ *htv*)

4.22.3.9 void destroy_pseudo_cluster (pseudo_cluster ∗ *ps*)

4.22.3.10 void destroy_var_undo_data (var_undo_data ∗ *vud*)

4.22.3.11 void do_something_with_solution (gbtd_system ∗ *gs*)

4.22.3.12 int filter_gvars (gbtd_system ∗ *gs,* int *skip_if_all_revised*)

4.22.3.13 void find_gvar (gbtd_system ∗ *gs*)

4.22.3.14 void find_gvar_wtn (gbtd_system ∗ *gs,* int *backwards*)

4.22.3.15 void generate_pseudo_tree (gbtd_system ∗ *gs*)

4.22.3.16 double has_good (pseudo_cluster ∗ *psc*)

4.22.3.17 int has_nogood (pseudo_cluster ∗ *psc*)

4.22.3.18 gbtd_system∗ init_gbtd (llist ∗ *vars*, int *allSol*, options *system*, set ∗ *queue*, set ∗∗ *queues*, int *total_clusters*, decomposed_tree ∗∗ *treens*, comb_data ∗ *cd*, int *revision_time*, int ∗ *tuple_positions*, int ∗ *tuple_position_htid*, active_tuple ∗ *ac*, int *count_on_m_s*)

4.22.3.19 void init_goods_nogoods (pseudo_cluster ∗ *psc*)

4.22.3.20 void move_to_previous_var (gbtd_system ∗ *gs*)

4.22.3.21 gbtd_system∗ new_gbtd_system (void)

4.22.3.22 htv_undo_data∗ new_htv_undo_data (htvar ∗ *htv*, gbtd_system ∗ *gs*)

4.22.3.23 htvar∗ new_htvar (htable ∗ *ht*, int *id*, llist_node ∗ *chosen_value*)

4.22.3.24 pseudo_cluster∗ new_pseudo_cluster (void)

4.22.3.25 var_undo_data∗ new_var_undo_data (variable ∗ *var*)

4.22.3.26 int no_unassigned_cover_variable (gbtd_system ∗ *gs*)

4.22.3.27 void print_gbtd_system (gbtd_system ∗ *gs*)

4.22.3.28 int relation_var_tuple_stamp (gbtd_system ∗ *gs*, htvar ∗ *htv*, int ∗ *tuple*, int *first*, int *marked*)

4.22.3.29 void reset_gvar (gbtd_system ∗ *gs*, void ∗ *gvar*)

4.22.3.30 int run_gbtd (gbtd_system ∗ *gs*)

4.22.3.31 int run_gbtd_wtn (gbtd_system ∗ *gs*)

4.22.3.32 void set_good_nogood (pseudo_cluster ∗ *psc*)

4.22.3.33 pseudo_cluster∗ sibling (pseudo_cluster ∗ *pc*)

4.22.3.34 int unassign_subtree (gbtd_system ∗ *gs*)

4.22.3.35 void undo_htv_assignment (htv_undo_data ∗ *vud*, gbtd_system ∗ *gs*)

4.23 globals.c File Reference

```
#include "globals.h"
#include "utils.h"
#include "solution.h"
```

Functions

- int get_partition_id ()
- void into_components_problem (main_structure ∗m_s)
- void set_m_s_basedon_this (main_structure ∗newms, main_structure ∗m_s)
- void add_single_variable_component (int varnumber, main_structure ∗m_s)
- void add_component (int vars, int rels, main_structure ∗m_s, llist ∗partition)

4.23.1 Function Documentation

4.23.1.1 void add_component (int *vars,* int *rels,* main_structure ∗ *m_s,* llist ∗ *partition*)

4.23.1.2 void add_single_variable_component (int *varnumber,* main_structure ∗ *m_s*)

4.23.1.3 int get_partition_id (void)

4.23.1.4 void into_components_problem (main_structure ∗ *m_s*)

4.23.1.5 void set_m_s_basedon_this (main_structure ∗ *newms,* main_structure ∗ *m_s*)

4.24 globals.h File Reference

```
#include <stdlib.h>
#include "variables.h"
#include "constraints.h"
#include "relations.h"
#include "variable_tracking.h"
#include "set.h"
#include <time.h>
```

Classes

- struct main_structure

Typedefs

- typedef enum options options
- typedef struct main_structure main_structure

Enumerations

- enum options {
EMPTY, PAIR_INDEX_FC, PAIR_INDEX_BC, PAIR_INDEX_ALL_SOL,
PAIR_INDEX_AUTO, MULTI_TUPLE_INDEX, SKIP_INSOL_TUPLES, OLD_STUFF,
MIN_FILL, LIB, LIB_DUAL, MIN_FILL_DUAL,
RS_GAC, MIN_FILL_LIB, NOSPLITSEP, GLOBALSEP,
COMBSEP, MBESEP, MAXCLIQUESEP, MAXCLIQUESEP2,
MAXCLIQUESEP_PP, BINSEP, BINSEP_PP, ALLSCOPESSP,
SPLTSEP, ALLSOLSEP, ONESOLSEP, RMC,
RMCSEP, RMCSEP_SPLT, RMCSEP_SPLT_ALL_SOL, RSTAR,
BTD, MBE, MBErs, CDATA_TREE_SEARCH,
CDATA_JOIN, CDATA_IGNORE, CDATA_RS_COMB, RS2TREE_SINGLE_GRAPH,
RS2TREE_PER_CLUSTER, RS2TREE_PER_CLUSTER_SINGLEQ, RS2TREE_PER_CLUSTER_2, RS2TREE_PER_CLUSTER_ALL_1SOL,
RS2TREE_SINGLE_DISTRIBUTED_GRAPH, RS2TREE_PER_PARENT_CHILD, RS2TREE_TUPLE_SEARCH, SCOPE_SUBSET_FIT,
SCOPE_SUBSET_FIT_NOT_IN_DESC, SCOPE_PROJECT_FIT, SCOPE_PROJECT_BIGGEST_FIT, ONE,
ALL, AUTO, DEFAULT_QUEUE, MANAGED_QUEUE_1,
MANAGED_QUEUE_12, MANAGED_QUEUE_122, MANAGED_QUEUE_13, MANAGED_QUEUE_14,
MANAGED_QUEUE_2, MANAGED_QUEUE_d, MANAGED_QUEUE_r, MANAGED_QUEUE_SINGLEQ,
GBTD_SYSTEM_VARS_SINGLE_CLUSTER, GBTD_SYSTEM_VARS_TD, GBTD_SYSTEM_RELS_SING-

LE_CLUSTER, USE_BTD,
USE_BTD_WTN, BINCLQ, PROJ, COLLECT_PRINT }

Functions

- int get_partition_id (void)
- void check_ok (main_structure ∗m_s)
- void check_ok2 (variables ∗vars)
- void into_components_problem (main_structure ∗m_s)
- void add_component (int vars, int rels, main_structure ∗m_s, llist ∗partition)
- void set_m_s_basedon_this (main_structure ∗newms, main_structure ∗m_s)
- void add_single_variable_component (int varnumber, main_structure ∗m_s)

4.24.1 Detailed Description

Represents the global parameters controlling the program.

4.24.2 Typedef Documentation

4.24.2.1 typedef struct **main_structure main_structure**

4.24.2.2 typedef enum **options options**

4.24.3 Enumeration Type Documentation

4.24.3.1 enum **options**

Enumerator

EMPTY

PAIR_INDEX_FC

PAIR_INDEX_BC

PAIR_INDEX_ALL_SOL

PAIR_INDEX_AUTO

MULTI_TUPLE_INDEX

SKIP_INSOL_TUPLES

OLD_STUFF

MIN_FILL

LIB

LIB_DUAL

MIN_FILL_DUAL

RS_GAC

MIN_FILL_LIB

NOSPLITSEP

GLOBALSEP

COMBSEP

MBESEP

MAXCLIQUESEP

MAXCLIQUESEP2

MAXCLIQUESEP_PP

BINSEP

BINSEP_PP

ALLSCOPESSP

SPLTSEP

ALLSOLSEP

ONESOLSEP

RMC

RMCSEP

RMCSEP_SPLT

RMCSEP_SPLT_ALL_SOL

RSTAR

BTD

MBE

MBErs

CDATA_TREE_SEARCH

CDATA_JOIN

CDATA_IGNORE

CDATA_RS_COMB

RS2TREE_SINGLE_GRAPH

RS2TREE_PER_CLUSTER

RS2TREE_PER_CLUSTER_SINGLEQ

RS2TREE_PER_CLUSTER_2

RS2TREE_PER_CLUSTER_ALL_1SOL

RS2TREE_SINGLE_DISTRIBUTED_GRAPH

RS2TREE_PER_PARENT_CHILD

RS2TREE_TUPLE_SEARCH

SCOPE_SUBSET_FIT

SCOPE_SUBSET_FIT_NOT_IN_DESC

SCOPE_PROJECT_FIT

SCOPE_PROJECT_BIGGEST_FIT

ONE

ALL

AUTO

DEFAULT_QUEUE

MANAGED_QUEUE_1

MANAGED_QUEUE_12

MANAGED_QUEUE_122

MANAGED_QUEUE_13

MANAGED_QUEUE_14

MANAGED_QUEUE_2

MANAGED_QUEUE_d

MANAGED_QUEUE_r

MANAGED_QUEUE_SINGLEQ

GBTD_SYSTEM_VARS_SINGLE_CLUSTER

GBTD_SYSTEM_VARS_TD

GBTD_SYSTEM_RELS_SINGLE_CLUSTER

USE_BTD

USE_BTD_WTN

BINCLQ

PROJ

COLLECT_PRINT

4.24.4 Function Documentation

4.24.4.1 void add_component (int *vars*, int *rels*, main_structure ∗ *m_s*, llist ∗ *partition*)

4.24.4.2 void add_single_variable_component (int *varnumber*, main_structure ∗ *m_s*)

4.24.4.3 void check_ok (main_structure ∗ *m_s*)

4.24.4.4 void check_ok2 (variables ∗ *vars*)

4.24.4.5 int get_partition_id (void)

4.24.4.6 void into_components_problem (main_structure ∗ *m_s*)

4.24.4.7 void set_m_s_basedon_this (main_structure ∗ *newms*, main_structure ∗ *m_s*)

4.25 hashmap.c File Reference

```
#include "hashmap.h"
#include "utils.h"
#include <math.h>
#include <stdio.h>
```

Functions

- hashmap ∗ new_hashmap (int size, int width, int ∗col_pos, int ∗max, int single)
- void destroy_hm (hashmap ∗hm)
- int HashDivide (KEYTYPE k, int m)
- int HashMultiply (KEYTYPE k, int m)
- int DoubleProbing (KEYTYPE k, int i, int size)
- int tupple_match (int ∗ta, int ∗tb, hashmap ∗hm, int ∗col_pos)
- void insert (int ∗tupple, hashmap ∗hm)
- llist ∗ get (int ∗tupple, hashmap ∗hm, int ∗col_pos)
- KEYTYPE get_key (int ∗tuple, hashmap ∗hm, int ∗col_pos)
- void hm_get_iterator (hashmap ∗hm, int ∗tuple, int ∗col_pos)
- int ∗ hm_get_next (hashmap ∗hm)

4.25.1 Function Documentation

4.25.1.1 void destroy_hm (hashmap ∗ *hm*)

4.25.1.2 int DoubleProbing (KEYTYPE *k*, int *i*, int *size*)

4.25.1.3 llist∗ get (int ∗ *tupple,* hashmap ∗ *hm,* int ∗ *col_pos*)

4.25.1.4 KEYTYPE get_key (int ∗ *tuple,* hashmap ∗ *hm,* int ∗ *col_pos*)

4.25.1.5 int HashDivide (KEYTYPE *k,* int *m*)

4.25.1.6 int HashMultiply (KEYTYPE *k,* int *m*)

4.25.1.7 void hm_get_iterator (hashmap ∗ *hm,* int ∗ *tuple,* int ∗ *col_pos*)

4.25.1.8 int∗ hm_get_next (hashmap ∗ *hm*)

4.25.1.9 void insert (int ∗ *tupple,* hashmap ∗ *hm*)

4.25.1.10 hashmap∗ new_hashmap (int *size,* int *width,* int ∗ *col_pos,* int ∗ *max,* int *single*)

4.25.1.11 int tupple_match (int ∗ *ta,* int ∗ *tb,* hashmap ∗ *hm,* int ∗ *col_pos*)

4.26 hashmap.h File Reference

```
#include "llist.h"
```

Classes

- struct hashmap

Macros

- #define KEYTYPE long long

Typedefs

- typedef struct hashmap hashmap

Functions

- hashmap ∗ new_hashmap (int size, int width, int ∗col_pos, int ∗max, int single)
- void destroy_hm (hashmap ∗hm)
- int HashDivide (KEYTYPE k, int m)
- int HashMultiply (KEYTYPE k, int m)
- int DoubleProbing (KEYTYPE k, int i, int size)
- void insert (int ∗tupple, hashmap ∗hm)
- llist ∗ get (int ∗tupple, hashmap ∗hm, int ∗col_pos)
- KEYTYPE get_key (int ∗tuple, hashmap ∗hm, int ∗col_pos)
- int tupple_match (int ∗ta, int ∗tb, hashmap ∗hm, int ∗col_pos)
- void get_iterator (hashmap ∗hm, int ∗tuple, int ∗col_pos)
- int ∗ hm_get_next (hashmap ∗hm)
- void hm_get_iterator (hashmap ∗hm, int ∗tuple, int ∗col_pos)

4.26.1 Detailed Description

Provides a hashmap implementation.

4.26.2 Macro Definition Documentation

4.26.2.1 #define KEYTYPE long long

4.26.3 Typedef Documentation

4.26.3.1 typedef struct **hashmap hashmap**

4.26.4 Function Documentation

4.26.4.1 void destroy_hm (**hashmap** ∗ *hm*)

4.26.4.2 int DoubleProbing (**KEYTYPE** *k,* int *i,* int *size*)

4.26.4.3 **llist**∗ get (int ∗ *tupple,* **hashmap** ∗ *hm,* int ∗ *col_pos*)

4.26.4.4 void get_iterator (**hashmap** ∗ *hm,* int ∗ *tuple,* int ∗ *col_pos*)

4.26.4.5 **KEYTYPE** get_key (int ∗ *tuple,* **hashmap** ∗ *hm,* int ∗ *col_pos*)

4.26.4.6 int HashDivide (**KEYTYPE** *k,* int *m*)

4.26.4.7 int HashMultiply (**KEYTYPE** *k,* int *m*)

4.26.4.8 void hm_get_iterator (**hashmap** ∗ *hm,* int ∗ *tuple,* int ∗ *col_pos*)

4.26.4.9 int∗ hm_get_next (**hashmap** ∗ *hm*)

4.26.4.10 void insert (int ∗ *tupple,* **hashmap** ∗ *hm*)

4.26.4.11 **hashmap**∗ new_hashmap (int *size,* int *width,* int ∗ *col_pos,* int ∗ *max,* int *single*)

4.26.4.12 int tupple_match (int ∗ *ta,* int ∗ *tb,* **hashmap** ∗ *hm,* int ∗ *col_pos*)

4.27 hashtable.c File Reference

```
#include <stdio.h>
#include <string.h>
#include <math.h>
#include "hashtable.h"
#include "hashmap.h"
#include "sort.h"
#include "utils.h"
#include "globals.h"
#include "solution.h"
#include "tree_map.h"
#include "rb_tree.h"
```

Functions

- int ∗ get_int_pair ()
- htable ∗ new_htable (int id, const char ∗name, int width, int hight, htable ∗prev, char ∗∗col_names, int join_cutoff, int is_first, int ∗vars)

- htable ∗ new_htable_replace (int id, const char ∗name, int width, int hight, htable ∗prev, char ∗∗col_names, int join_cutoff, int is_first, int ∗vars, int replace)
- void add_tuple_list (htable ∗ht)
- llist ∗ get_ht_iterator (htable ∗ht)
- void clear_deleted_rows (htable ∗ht)
- char ∗∗ copy_col_names (char ∗∗col_names, int size)
- void destroy_histogram (histogram ∗histo)
- histogram ∗ new_histogram (int columns)
- int compare_col_name (void ∗value, void ∗bodyb)
- int compare_col_card (void ∗value, void ∗bodyb)
- int estimate_join_size (htable ∗hta, htable ∗htb)
- int estimate_join_size_noproject (htable ∗hta, htable ∗htb)
- double ceil (double x)
- int estimate_project_size (htable ∗hta, llist ∗project_cols)
- int estimate_join_project_size (htable ∗hta, htable ∗htb, int project)
- void build_histogram (htable ∗table)
- void rebuild_histogram (htable ∗table)
- void update_histogram (htable ∗table)
- htable ∗ new_clustered_htable (int id, char ∗name, int width, int hight, char ∗∗col_names, int join_cutoff, int ∗vars)
- int ∗ get_tuple (int row, htable ∗ht)
- double get_tuple_value (int row, htable ∗ht)
- void set_tuple_value (double value, int row, htable ∗ht)
- int get_value (int row, int column, htable ∗ht)
- int get_value_sort (void ∗ht, int row, int column)
- void set_value_sort (void ∗vht, int row, int column, int value)
- void destroy_htable (htable ∗ht)
- void destroy_htable_o (htable ∗ht, int kill_col_names)
- char ∗∗ new_names (int size)
- void add_name (char ∗name, char ∗∗names, int pos)
- htable ∗ add_tuple (int ∗tuple, htable ∗ht)
- void print_ht_mysql (htable ∗ht)
- void print_ht_h (htable ∗ht, int short_name)
- void print_ht_line (htable ∗ht)
- void print_ht (htable ∗ht, int short_name)
- void fprint_ht (FILE ∗file, htable ∗ht, int short_name)
- void replace_data (htable ∗data, htable ∗ht)
- int have_value (int x)
- htable ∗ cluster (int col1, int col2, htable ∗ht, int no_destroy)
- htable ∗ cluster_nosort (htable ∗ht, int no_destroy)
- htable ∗ cluster_sort (int col1, int col2, htable ∗ht, int no_destroy, int sort)
- int ∗ col_names_to_vars (llist ∗varnames)
- htable ∗ load_from_tuples (int id, char ∗cons_name, char ∗∗col_names, tuple ∗tuples, int ∗vars, int header_only, constraint ∗cons)
- htable ∗ load_from_conflict_tuples (int id, char ∗cons_name, char ∗∗col_names, tuple ∗tuples, llist ∗cons_var_domains, int ∗vars)
- htable ∗ convert_conflict_to_support4 (htable ∗conf)
- htable ∗ convert_conflict_to_supportn (htable ∗conf)
- int ∗ init_get_next_tuple (htable ∗ht)
- int get_next_tuple (htable ∗ht, int ∗current_tuple)
- void convert_conflict_to_supportn_rec (htable ∗conf, htable ∗support, int ∗current_tuple_pos, int ∗t, int v)
- htable ∗ convert_conflict_to_support3 (htable ∗conf)
- htable ∗ convert_conflict_to_support2 (htable ∗conf)
- int get_int_common_col_count (int ∗var1, int ∗var2, int ∗vmap)
- int ∗ get_projection_scope (llist ∗var1, int ∗var2, int ∗vmap)

- void destroy_int_common_cols_list (llist ∗l)
- llist ∗ get_int_common_cols_list (llist ∗var1, int ∗var2, int ∗vmap)
- llist ∗ get_int_common_cols (int ∗var1, int ∗var2, int ∗vmap)
- llist ∗ get_int_common_cols_vars_list (int ∗var1, int ∗var2, llist ∗varList, int ∗vmap)
- int has_int_common_cols (int ∗var1, int ∗var2, int ∗vmap)
- common_var ∗ new_common_var (int a, int b)
- int do_compare (htable ∗hta, htable ∗htb, llist ∗common, int i, int j)
- int do_match (htable ∗hta, htable ∗htb, llist ∗common, int i, int j)
- int do_match_next (htable ∗hta, llist ∗common, int i, int hta_htb)
- void merge_tupple (htable ∗hta, htable ∗htb, int ∗tc, int ∗ta, int ∗tb, int ∗common_map_htb)
- void merge (htable ∗hta, htable ∗htb, int ∗tc, int x, int y, int ∗common_map_htb)
- int compare_clustered (void ∗ba, void ∗bb)
- htable ∗ hash_join_tables (htable ∗hta, htable ∗htb)
- htable ∗ project_join_tables (htable ∗hta, htable ∗htb)
- htable ∗ join_tables_new (htable ∗htap, htable ∗htbp)
- htable ∗ join_tables (htable ∗∗htap, htable ∗∗htbp, int limit, htable ∗joined_table, int no_destroy)
- htable ∗ join_tables_w_suspend (htable ∗∗htap, htable ∗∗htbp, int limit, htable ∗joined_table, int no_destroy)
- llist ∗ get_common_cols (htable ∗hta, htable ∗htb)
- int match_tuple_a_tuple_b (int ∗a, int ∗b, llist ∗common_var_array_list)
- htable ∗ distinct (htable ∗ht)
- htable ∗ project_table (htable ∗ht, llist ∗cols)
- htable ∗ project_distinct_table (htable ∗ht, llist ∗cols, int distinct)
- htable ∗ purge_htable (htable ∗ht)
- htable ∗ copy_htable (htable ∗ht)
- hashmap ∗ init_hashmap (htable ∗ht, llist ∗cols)
- hashmap ∗ get_hashmap_wd (htable ∗ht, llist ∗cols)
- htable ∗ hash_project_distinct_table (htable ∗ht, llist ∗cols, int distinct)
- htable ∗ prepare_pipelined_join (htable ∗∗htap, htable ∗∗htbp, htable ∗joined_table, int no_destroy)
- void filter_table (htable ∗ht, alive_table ∗at)
- void update_alive (alive_table ∗at, htable ∗ht)
- void track_alive (htable ∗ht)
- void delete_tuple (htable ∗ht, int tuple)
- void un_delete_tuple (htable ∗ht, int tuple)
- void un_delete_tuple_old (htable ∗ht, int tuple)
- int match_tuple_single_ht (int ∗tuple, llist ∗commons, int twisted, htable ∗ht)
- int get_next_alive_index (int t, htable ∗ht)
- int ∗ get_next (htable ∗ht)
- int get_it_value (void ∗a, int i, int c)
- void ∗ get_it_object (void ∗a, int i, int c)
- void set_it_object (void ∗a, int i, int c, void ∗obj)
- void sort_ht_iterator (htable ∗ht)
- tuple_ids ∗ new_tuple_ids (int ∗tuple)
- tuple_ids ∗ merge_two_tuples_ids (tuple_ids ∗ta, tuple_ids ∗tb, llist ∗comms)
- void free_tuple_ids (tuple_ids ∗ti)
- htable ∗ create_fill_htable (llist ∗varptrs, const char ∗name)
- void update_current_domains (htable ∗ht, light_stack ∗undo_value_stack)
- void filter_table_with_current_domains (htable ∗ht)
- double get_domain_product (htable ∗ht)
- void resetset_variable_hashtables (llist ∗htables)
- void set_variable_hashtables (llist ∗htables)
- void set_variable_hashtable (htable ∗ht)
- void filter_domains_with_unary_constraints (variables ∗vars)
- int compare_scope (const void ∗o1, const void ∗o2)
- int compare_scope_buble (void ∗o1, void ∗o2)

Variables

- int current_markgeneration = 0

4.27.1 Function Documentation

4.27.1.1 void add_name (char ∗ *name,* char ∗∗ *names,* int *pos*)

4.27.1.2 htable∗ add_tuple (int ∗ *tuple,* htable ∗ *ht*)

4.27.1.3 void add_tuple_list (htable ∗ *ht*)

4.27.1.4 void build_histogram (htable ∗ *table*)

4.27.1.5 double ceil (double *x*)

4.27.1.6 void clear_deleted_rows (htable ∗ *ht*)

4.27.1.7 htable∗ cluster (int *col1,* int *col2,* htable ∗ *ht,* int *no_destroy*)

4.27.1.8 htable∗ cluster_nosort (htable ∗ *ht,* int *no_destroy*)

4.27.1.9 htable∗ cluster_sort (int *col1,* int *col2,* htable ∗ *ht,* int *no_destroy,* int *sort*)

4.27.1.10 int∗ col_names_to_vars (llist ∗ *varnames*)

4.27.1.11 int compare_clustered (void ∗ *ba,* void ∗ *bb*)

4.27.1.12 int compare_col_card (void ∗ *value,* void ∗ *bodyb*)

4.27.1.13 int compare_col_name (void ∗ *value,* void ∗ *bodyb*)

4.27.1.14 int compare_scope (const void ∗ *o1,* const void ∗ *o2*)

4.27.1.15 int compare_scope_buble (void ∗ *o1,* void ∗ *o2*)

4.27.1.16 htable∗ convert_conflict_to_support2 (htable ∗ *conf*)

4.27.1.17 htable∗ convert_conflict_to_support3 (htable ∗ *conf*)

4.27.1.18 htable∗ convert_conflict_to_support4 (htable ∗ *conf*)

4.27.1.19 htable∗ convert_conflict_to_supportn (htable ∗ *conf*)

4.27.1.20 void convert_conflict_to_supportn_rec (htable ∗ *conf,* htable ∗ *support,* int ∗ *current_tuple_pos,* int ∗ *t,* int *v*)

4.27.1.21 char∗∗ copy_col_names (char ∗∗ *col_names,* int *size*)

4.27.1.22 htable∗ copy_htable (htable ∗ *ht*)

4.27.1.23 htable∗ create_fill_htable (llist ∗ *varptrs,* const char ∗ *name*)

4.27.1.24 void delete_tuple (htable ∗ *ht,* int *tuple*)

4.27.1.25 void destroy_histogram (histogram ∗ *histo*)

4.27.1.26 void destroy_htable (**htable** ∗ *ht*)

4.27.1.27 void destroy_htable_o (**htable** ∗ *ht,* int *kill_col_names*)

4.27.1.28 void destroy_int_common_cols_list (**llist** ∗ *l*)

4.27.1.29 **htable**∗ distinct (**htable** ∗ *ht*)

4.27.1.30 int do_compare (**htable** ∗ *hta,* **htable** ∗ *htb,* **llist** ∗ *common,* int *i,* int *j*)

4.27.1.31 int do_match (**htable** ∗ *hta,* **htable** ∗ *htb,* **llist** ∗ *common,* int *i,* int *j*)

4.27.1.32 int do_match_next (**htable** ∗ *hta,* **llist** ∗ *common,* int *i,* int *hta_htb*)

4.27.1.33 int estimate_join_project_size (**htable** ∗ *hta,* **htable** ∗ *htb,* int *project*)

4.27.1.34 int estimate_join_size (**htable** ∗ *hta,* **htable** ∗ *htb*)

4.27.1.35 int estimate_join_size_noproject (**htable** ∗ *hta,* **htable** ∗ *htb*)

4.27.1.36 int estimate_project_size (**htable** ∗ *hta,* **llist** ∗ *project_cols*)

4.27.1.37 void filter_domains_with_unary_constraints (**variables** ∗ *vars*)

4.27.1.38 void filter_table (**htable** ∗ *ht,* alive_table ∗ *at*)

4.27.1.39 void filter_table_with_current_domains (**htable** ∗ *ht*)

4.27.1.40 void fprint_ht (FILE ∗ *file,* **htable** ∗ *ht,* int *short_name*)

4.27.1.41 void free_tuple_ids (**tuple_ids** ∗ *ti*)

4.27.1.42 **llist**∗ get_common_cols (**htable** ∗ *hta,* **htable** ∗ *htb*)

4.27.1.43 double get_domain_product (**htable** ∗ *ht*)

4.27.1.44 **hashmap**∗ get_hashmap_wd (**htable** ∗ *ht,* **llist** ∗ *cols*)

4.27.1.45 **llist**∗ get_ht_iterator (**htable** ∗ *ht*)

4.27.1.46 int get_int_common_col_count (int ∗ *var1,* int ∗ *var2,* int ∗ *vmap*)

4.27.1.47 **llist**∗ get_int_common_cols (int ∗ *var1,* int ∗ *var2,* int ∗ *vmap*)

4.27.1.48 **llist**∗ get_int_common_cols_list (**llist** ∗ *var1,* int ∗ *var2,* int ∗ *vmap*)

4.27.1.49 **llist**∗ get_int_common_cols_vars_list (int ∗ *var1,* int ∗ *var2,* **llist** ∗ *varList,* int ∗ *vmap*)

4.27.1.50 int∗ get_int_pair (void)

4.27.1.51 void∗ get_it_object (void ∗ *a,* int *i,* int *c*)

4.27.1.52 int get_it_value (void ∗ *a,* int *i,* int *c*)

4.27.1.53 int∗ get_next (**htable** ∗ *ht*)

4.27.1.54 int get_next_alive_index (int t, htable * ht)

4.27.1.55 int get_next_tuple (htable * ht, int * current_tuple)

4.27.1.56 int* get_projection_scope (llist * var1, int * var2, int * vmap)

4.27.1.57 int* get_tuple (int row, htable * ht)

4.27.1.58 double get_tuple_value (int row, htable * ht)

4.27.1.59 int get_value (int row, int column, htable * ht)

4.27.1.60 int get_value_sort (void * ht, int row, int column)

4.27.1.61 int has_int_common_cols (int * var1, int * var2, int * vmap)

4.27.1.62 htable* hash_join_tables (htable * hta, htable * htb)

4.27.1.63 htable* hash_project_distinct_table (htable * ht, llist * cols, int distinct)

4.27.1.64 int have_value (int x)

4.27.1.65 int* init_get_next_tuple (htable * ht)

4.27.1.66 hashmap* init_hashmap (htable * ht, llist * cols)

4.27.1.67 htable* join_tables (htable ** htap, htable ** htbp, int limit, htable * joined_table, int no_destroy)

4.27.1.68 htable* join_tables_new (htable * htap, htable * htbp)

4.27.1.69 htable* join_tables_w_suspend (htable ** htap, htable ** htbp, int limit, htable * joined_table, int no_destroy)

4.27.1.70 htable* load_from_conflict_tuples (int id, char * cons_name, char ** col_names, tuple * tuples, llist * cons_var_domains, int * vars)

4.27.1.71 htable* load_from_tuples (int id, char * cons_name, char ** col_names, tuple * tuples, int * vars, int header_only, constraint * cons)

4.27.1.72 int match_tuple_a_tuple_b (int * a, int * b, llist * common_var_array_list)

4.27.1.73 int match_tuple_single_ht (int * tuple, llist * commons, int twisted, htable * ht)

4.27.1.74 void merge (htable * hta, htable * htb, int * tc, int x, int y, int * common_map_htb)

4.27.1.75 void merge_tupple (htable * hta, htable * htb, int * tc, int * ta, int * tb, int * common_map_htb)

4.27.1.76 tuple_ids* merge_two_tuples_ids (tuple_ids * ta, tuple_ids * tb, llist * comms)

4.27.1.77 htable* new_clustered_htable (int id, char * name, int width, int hight, char ** col_names, int join_cutoff, int * vars)

4.27.1.78 common_var* new_common_var (int a, int b)

4.27.1.79 histogram* new_histogram (int columns)

4.27.1.80 htable∗ new_htable (int *id,* const char ∗ *name,* int *width,* int *hight,* htable ∗ *prev,* char ∗∗ *col_names,* int *join_cutoff,* int *is_first,* int ∗ *vars*)

4.27.1.81 htable∗ new_htable_replace (int *id,* const char ∗ *name,* int *width,* int *hight,* htable ∗ *prev,* char ∗∗ *col_names,* int *join_cutoff,* int *is_first,* int ∗ *vars,* int *replace*)

4.27.1.82 char∗∗ new_names (int *size*)

4.27.1.83 tuple_ids∗ new_tuple_ids (int ∗ *tuple*)

4.27.1.84 htable∗ prepare_pipelined_join (htable ∗∗ *htap,* htable ∗∗ *htbp,* htable ∗ *joined_table,* int *no_destroy*)

4.27.1.85 void print_ht (htable ∗ *ht,* int *short_name*)

4.27.1.86 void print_ht_h (htable ∗ *ht,* int *short_name*)

4.27.1.87 void print_ht_line (htable ∗ *ht*)

4.27.1.88 void print_ht_mysql (htable ∗ *ht*)

4.27.1.89 htable∗ project_distinct_table (htable ∗ *ht,* llist ∗ *cols,* int *distinct*)

4.27.1.90 htable∗ project_join_tables (htable ∗ *hta,* htable ∗ *htb*)

4.27.1.91 htable∗ project_table (htable ∗ *ht,* llist ∗ *cols*)

4.27.1.92 htable∗ purge_htable (htable ∗ *ht*)

4.27.1.93 void rebuild_histogram (htable ∗ *table*)

4.27.1.94 void replace_data (htable ∗ *data,* htable ∗ *ht*)

4.27.1.95 void resetset_variable_hashtables (llist ∗ *htables*)

4.27.1.96 void set_it_object (void ∗ *a,* int *i,* int *c,* void ∗ *obj*)

4.27.1.97 void set_tuple_value (double *value,* int *row,* htable ∗ *ht*)

4.27.1.98 void set_value_sort (void ∗ *vht,* int *row,* int *column,* int *value*)

4.27.1.99 void set_variable_hashtable (htable ∗ *ht*)

4.27.1.100 void set_variable_hashtables (llist ∗ *htables*)

4.27.1.101 void sort_ht_iterator (htable ∗ *ht*)

4.27.1.102 void track_alive (htable ∗ *ht*)

4.27.1.103 void un_delete_tuple (htable ∗ *ht,* int *tuple*)

4.27.1.104 void un_delete_tuple_old (htable ∗ *ht,* int *tuple*)

4.27.1.105 void update_alive (alive_table ∗ *at,* htable ∗ *ht*)

4.27.1.106 void update_current_domains (htable ∗ *ht,* light_stack ∗ *undo_value_stack*)

4.27.1.107 void update_histogram (**htable** ∗ *table*)

4.27.2 Variable Documentation

4.27.2.1 int current_markgeneration = 0

4.28 hashtable.h File Reference

```
#include <stdio.h>
#include "relations.h"
#include "variable_tracking.h"
#include "constraints.h"
#include "hashmap.h"
#include "variables.h"
#include "globals.h"
```

Classes

- struct htable
- struct histogram
- struct common_var
- struct col_card
- struct tuple_ids

Macros

- #define DEF_HIGHT 4
- #define INF_HIGHT 999999999
- #define DELETED width+1
- #define TIMESTP width+2
- #define REVTIME width+3
- #define TUPMARK width+4
- #define ADD_FOR_PHYSICAL_WIDTH 6

Typedefs

- typedef struct htable htable
- typedef struct histogram histogram
- typedef struct common_var common_var
- typedef struct col_card col_card
- typedef struct tuple_ids tuple_ids

Functions

- int ∗ get_int_pair (void)
- int compare_col_card (void ∗value, void ∗bodyb)
- int compare_col_name (void ∗value, void ∗bodyb)
- histogram ∗ new_histogram (int columns)
- void build_histogram (htable ∗table)
- void rebuild_histogram (htable ∗table)
- void update_histogram (htable ∗table)

- int estimate_join_size_noproject (htable ∗hta, htable ∗htb)
- int estimate_project_size (htable ∗hta, llist ∗project_cols)
- int estimate_join_project_size (htable ∗hta, htable ∗htb, int project)
- htable ∗ hash_project_distinct_table (htable ∗ht, llist ∗cols, int distinct)
- int estimate_join_size (htable ∗ha, htable ∗hb)
- htable ∗ new_htable_replace (int id, const char ∗name, int width, int hight, htable ∗prev, char ∗∗col_names, int join_cutoff, int is_first, int ∗vars, int replace)
- htable ∗ new_htable (int id, const char ∗name, int width, int hight, htable ∗prev, char ∗∗col_names, int join_cutoff, int is_first, int ∗vars)
- char ∗∗ copy_col_names (char ∗∗col_names, int size)
- htable ∗ new_clustered_htable (int id, char ∗name, int width, int hight, char ∗∗col_names, int joi_cutoff, int ∗vars)
- int ∗ get_tuple (int row, htable ∗ht)
- double get_tuple_value (int row, htable ∗ht)
- void set_tuple_value (double value, int row, htable ∗ht)
- int get_value (int row, int column, htable ∗ht)
- void destroy_htable (htable ∗ht)
- void destroy_htable_o (htable ∗ht, int kill_col_names)
- char ∗∗ new_names (int size)
- void add_name (char ∗name, char ∗∗names, int pos)
- htable ∗ add_tuple (int ∗tuple, htable ∗ht)
- htable ∗ cluster_nosort (htable ∗ht, int no_destroy)
- htable ∗ cluster_sort (int col1, int col2, htable ∗ht, int no_destroy, int sort)
- htable ∗ cluster (int col1, int col2, htable ∗ht, int no_destroy)
- void print_ht_h (htable ∗ht, int short_name)
- void print_ht (htable ∗ht, int short_name)
- void fprint_ht (FILE ∗file, htable ∗ht, int short_name)
- void print_ht_mysql (htable ∗ht)
- htable ∗ load_from_tuples (int id, char ∗cons_name, char ∗∗col_names, tuple ∗tuples, int ∗vars, int header_only, constraint ∗cons)
- htable ∗ load_from_conflict_tuples (int id, char ∗cons_name, char ∗∗col_names, tuple ∗tuples, llist ∗cons_var_domains, int ∗vars)
- htable ∗ convert_conflict_to_support2 (htable ∗conf)
- htable ∗ convert_conflict_to_support3 (htable ∗conf)
- htable ∗ convert_conflict_to_support4 (htable ∗conf)
- htable ∗ project_distinct_table (htable ∗ht, llist ∗cols, int distinct)
- void replace_data (htable ∗data, htable ∗ht)
- htable ∗ project_table (htable ∗ht, llist ∗cols)
- htable ∗ project_join_tables (htable ∗hta, htable ∗htb)
- htable ∗ join_tables (htable ∗∗htap, htable ∗∗htbp, int limit, htable ∗joined_table, int no_destroy)
- htable ∗ hash_join_tables (htable ∗hta, htable ∗htb)
- htable ∗ join_tables_w_suspend (htable ∗∗htap, htable ∗∗htbp, int limit, htable ∗joined_table, int no_destroy)
- void destroy_int_common_cols_list (llist ∗l)
- llist ∗ get_common_cols (htable ∗hta, htable ∗htb)
- llist ∗ get_int_common_cols_list (llist ∗var1, int ∗var2, int ∗vmap)
- llist ∗ get_int_common_cols_vars_list (int ∗var1, int ∗var2, llist ∗varList, int ∗vmap)
- int ∗ get_projection_scope (llist ∗var1, int ∗var2, int ∗vmap)
- htable ∗ distinct (htable ∗ht)
- int get_value_sort (void ∗ht, int row, int column)
- void set_value_sort (void ∗vht, int row, int column, int value)
- int have_value (int x)
- int get_int_common_col_count (int ∗var1, int ∗var2, int ∗vmap)
- int has_int_common_cols (int ∗var1, int ∗var2, int ∗vmap)
- llist ∗ get_int_common_cols (int ∗var1, int ∗var2, int ∗vmap)
- common_var ∗ new_common_var (int a, int b)

- int do_compare (htable ∗hta, htable ∗htb, llist ∗common, int i, int j)
- int do_match (htable ∗hta, htable ∗htb, llist ∗common, int i, int j)
- int do_match_next (htable ∗hta, llist ∗common, int i, int hta_htb)
- int match_tuple_a_tuple_b (int ∗a, int ∗b, llist ∗common_var_array_list)
- void merge (htable ∗hta, htable ∗htb, int ∗tc, int x, int y, int ∗common_map_htb)
- void merge_tupple (htable ∗hta, htable ∗htb, int ∗tc, int ∗ta, int ∗tb, int ∗common_map_htb)
- int compare_clustered (void ∗ba, void ∗bb)
- htable ∗ prepare_pipelined_join (htable ∗∗htap, htable ∗∗htbp, htable ∗joined_table, int no_destroy)
- hashmap ∗ get_hashmap_wd (htable ∗ht, llist ∗cols)
- hashmap ∗ init_hashmap (htable ∗ht, llist ∗cols)
- void update_alive (alive_table ∗at, htable ∗ht)
- void filter_table (htable ∗ht, alive_table ∗at)
- void track_alive (htable ∗ht)
- void clear_deleted_rows (htable ∗ht)
- void destroy_histogram (histogram ∗histo)
- void delete_tuple (htable ∗ht, int tuple)
- void un_delete_tuple_old (htable ∗ht, int tuple)
- void un_delete_tuple (htable ∗ht, int tuple)
- int match_tuple_single_ht (int ∗tuple, llist ∗commons, int twisted, htable ∗ht)
- int ∗ get_next (htable ∗ht)
- void add_tuple_list (htable ∗ht)
- llist ∗ get_ht_iterator (htable ∗ht)
- int get_it_value (void ∗a, int i, int c)
- void ∗ get_it_object (void ∗a, int i, int c)
- void set_it_object (void ∗a, int i, int c, void ∗obj)
- void sort_ht_iterator (htable ∗ht)
- int get_next_alive_index (int t, htable ∗ht)
- int ∗ col_names_to_vars (llist ∗varnames)
- tuple_ids ∗ new_tuple_ids (int ∗tuple)
- void free_tuple_ids (tuple_ids ∗ti)
- tuple_ids ∗ merge_two_tuples_ids (tuple_ids ∗ta, tuple_ids ∗tb, llist ∗comms)
- void convert_conflict_to_supportn_rec (htable ∗conf, htable ∗support, int ∗current_tuple, int ∗t, int v)
- htable ∗ convert_conflict_to_supportn (htable ∗conf)
- htable ∗ create_fill_htable (llist ∗varptrs, const char ∗name)
- int ∗ init_get_next_tuple (htable ∗ht)
- int get_next_tuple (htable ∗ht, int ∗current_tuple)
- void update_current_domains (htable ∗ht, light_stack ∗undo_value_stack)
- void filter_table_with_current_domains (htable ∗ht)
- htable ∗ join_tables_new (htable ∗htap, htable ∗htbp)
- double get_domain_product (htable ∗ht)
- htable ∗ copy_htable (htable ∗ht)
- htable ∗ purge_htable (htable ∗ht)
- void print_ht_line (htable ∗ht)
- void set_variable_hashtables (llist ∗htables)
- void set_variable_hashtable (htable ∗ht)
- void resetset_variable_hashtables (llist ∗htables)
- void filter_domains_with_unary_constraints (variables ∗vars)
- int compare_scope (const void ∗o1, const void ∗o2)
- int compare_scope_buble (void ∗o1, void ∗o2)

4.28.1 Detailed Description

Provides the data structures for representing the relations of the constraints and methods for constructing and manipulating them.

4.28.2 Macro Definition Documentation

4.28.2.1 #define ADD_FOR_PHYSICAL_WIDTH 6

4.28.2.2 #define DEF_HIGHT 4

4.28.2.3 #define DELETED width+1

4.28.2.4 #define INF_HIGHT 999999999

4.28.2.5 #define REVTIME width+3

4.28.2.6 #define TIMESTP width+2

4.28.2.7 #define TUPMARK width+4

4.28.3 Typedef Documentation

4.28.3.1 typedef struct col_card col_card

4.28.3.2 typedef struct common_var common_var

4.28.3.3 typedef struct histogram histogram

4.28.3.4 typedef struct htable htable

4.28.3.5 typedef struct tuple_ids tuple_ids

4.28.4 Function Documentation

4.28.4.1 void add_name (char ∗ *name,* char ∗∗ *names,* int *pos*)

4.28.4.2 htable∗ add_tuple (int ∗ *tuple,* htable ∗ *ht*)

4.28.4.3 void add_tuple_list (htable ∗ *ht*)

4.28.4.4 void build_histogram (htable ∗ *table*)

4.28.4.5 void clear_deleted_rows (htable ∗ *ht*)

4.28.4.6 htable∗ cluster (int *col1,* int *col2,* htable ∗ *ht,* int *no_destroy*)

4.28.4.7 htable∗ cluster_nosort (htable ∗ *ht,* int *no_destroy*)

4.28.4.8 htable∗ cluster_sort (int *col1,* int *col2,* htable ∗ *ht,* int *no_destroy,* int *sort*)

4.28.4.9 int∗ col_names_to_vars (llist ∗ *varnames*)

4.28.4.10 int compare_clustered (void ∗ *ba,* void ∗ *bb*)

4.28.4.11 int compare_col_card (void ∗ *value,* void ∗ *bodyb*)

4.28.4.12 int compare_col_name (void ∗ *value,* void ∗ *bodyb*)

4.28.4.13 int compare_scope (const void ∗ *o1,* const void ∗ *o2*)

4.28.4.14 int compare_scope_buble (void * *o1*, void * *o2*)

4.28.4.15 htable* convert_conflict_to_support2 (htable * *conf*)

4.28.4.16 htable* convert_conflict_to_support3 (htable * *conf*)

4.28.4.17 htable* convert_conflict_to_support4 (htable * *conf*)

4.28.4.18 htable* convert_conflict_to_supportn (htable * *conf*)

4.28.4.19 void convert_conflict_to_supportn_rec (htable * *conf*, htable * *support*, int * *current_tuple*, int * *t*, int *v*)

4.28.4.20 char** copy_col_names (char ** *col_names*, int *size*)

4.28.4.21 htable* copy_htable (htable * *ht*)

4.28.4.22 htable* create_fill_htable (llist * *varptrs*, const char * *name*)

4.28.4.23 void delete_tuple (htable * *ht*, int *tuple*)

4.28.4.24 void destroy_histogram (histogram * *histo*)

4.28.4.25 void destroy_htable (htable * *ht*)

4.28.4.26 void destroy_htable_o (htable * *ht*, int *kill_col_names*)

4.28.4.27 void destroy_int_common_cols_list (llist * *l*)

4.28.4.28 htable* distinct (htable * *ht*)

4.28.4.29 int do_compare (htable * *hta*, htable * *htb*, llist * *common*, int *i*, int *j*)

4.28.4.30 int do_match (htable * *hta*, htable * *htb*, llist * *common*, int *i*, int *j*)

4.28.4.31 int do_match_next (htable * *hta*, llist * *common*, int *i*, int *hta_htb*)

4.28.4.32 int estimate_join_project_size (htable * *hta*, htable * *htb*, int *project*)

4.28.4.33 int estimate_join_size (htable * *ha*, htable * *hb*)

4.28.4.34 int estimate_join_size_noproject (htable * *hta*, htable * *htb*)

4.28.4.35 int estimate_project_size (htable * *hta*, llist * *project_cols*)

4.28.4.36 void filter_domains_with_unary_constraints (variables * *vars*)

4.28.4.37 void filter_table (htable * *ht*, alive_table * *at*)

4.28.4.38 void filter_table_with_current_domains (htable * *ht*)

4.28.4.39 void fprint_ht (FILE * *file*, htable * *ht*, int *short_name*)

4.28.4.40 void free_tuple_ids (tuple_ids * *ti*)

4.28.4.41 llist* get_common_cols (htable * *hta*, htable * *htb*)

4.28.4.42 double get_domain_product (htable ∗ *ht*)

4.28.4.43 hashmap∗ get_hashmap_wd (htable ∗ *ht,* llist ∗ *cols*)

4.28.4.44 llist∗ get_ht_iterator (htable ∗ *ht*)

4.28.4.45 int get_int_common_col_count (int ∗ *var1,* int ∗ *var2,* int ∗ *vmap*)

4.28.4.46 llist∗ get_int_common_cols (int ∗ *var1,* int ∗ *var2,* int ∗ *vmap*)

4.28.4.47 llist∗ get_int_common_cols_list (llist ∗ *var1,* int ∗ *var2,* int ∗ *vmap*)

4.28.4.48 llist∗ get_int_common_cols_vars_list (int ∗ *var1,* int ∗ *var2,* llist ∗ *varList,* int ∗ *vmap*)

4.28.4.49 int∗ get_int_pair (void)

4.28.4.50 void∗ get_it_object (void ∗ *a,* int *i,* int *c*)

4.28.4.51 int get_it_value (void ∗ *a,* int *i,* int *c*)

4.28.4.52 int∗ get_next (htable ∗ *ht*)

4.28.4.53 int get_next_alive_index (int *t,* htable ∗ *ht*)

4.28.4.54 int get_next_tuple (htable ∗ *ht,* int ∗ *current_tuple*)

4.28.4.55 int∗ get_projection_scope (llist ∗ *var1,* int ∗ *var2,* int ∗ *vmap*)

4.28.4.56 int∗ get_tuple (int *row,* htable ∗ *ht*)

4.28.4.57 double get_tuple_value (int *row,* htable ∗ *ht*)

4.28.4.58 int get_value (int *row,* int *column,* htable ∗ *ht*)

4.28.4.59 int get_value_sort (void ∗ *ht,* int *row,* int *column*)

4.28.4.60 int has_int_common_cols (int ∗ *var1,* int ∗ *var2,* int ∗ *vmap*)

4.28.4.61 htable∗ hash_join_tables (htable ∗ *hta,* htable ∗ *htb*)

4.28.4.62 htable∗ hash_project_distinct_table (htable ∗ *ht,* llist ∗ *cols,* int *distinct*)

4.28.4.63 int have_value (int *x*)

4.28.4.64 int∗ init_get_next_tuple (htable ∗ *ht*)

4.28.4.65 hashmap∗ init_hashmap (htable ∗ *ht,* llist ∗ *cols*)

4.28.4.66 htable∗ join_tables (htable ∗∗ *htap,* htable ∗∗ *htbp,* int *limit,* htable ∗ *joined_table,* int *no_destroy*)

4.28.4.67 htable∗ join_tables_new (htable ∗ *htap,* htable ∗ *htbp*)

4.28.4.68 htable∗ join_tables_w_suspend (htable ∗∗ *htap,* htable ∗∗ *htbp,* int *limit,* htable ∗ *joined_table,* int *no_destroy*)

4.28.4.69 htable* load_from_conflict_tuples (int id, char * cons_name, char ** col_names, tuple * tuples, llist * cons_var_domains, int * vars)

4.28.4.70 htable* load_from_tuples (int id, char * cons_name, char ** col_names, tuple * tuples, int * vars, int header_only, constraint * cons)

4.28.4.71 int match_tuple_a_tuple_b (int * a, int * b, llist * common_var_array_list)

4.28.4.72 int match_tuple_single_ht (int * tuple, llist * commons, int twisted, htable * ht)

4.28.4.73 void merge (htable * hta, htable * htb, int * tc, int x, int y, int * common_map_htb)

4.28.4.74 void merge_tupple (htable * hta, htable * htb, int * tc, int * ta, int * tb, int * common_map_htb)

4.28.4.75 tuple_ids* merge_two_tuples_ids (tuple_ids * ta, tuple_ids * tb, llist * comms)

4.28.4.76 htable* new_clustered_htable (int id, char * name, int width, int hight, char ** col_names, int joi_cutoff, int * vars)

4.28.4.77 common_var* new_common_var (int a, int b)

4.28.4.78 histogram* new_histogram (int columns)

4.28.4.79 htable* new_htable (int id, const char * name, int width, int hight, htable * prev, char ** col_names, int join_cutoff, int is_first, int * vars)

4.28.4.80 htable* new_htable_replace (int id, const char * name, int width, int hight, htable * prev, char ** col_names, int join_cutoff, int is_first, int * vars, int replace)

4.28.4.81 char** new_names (int size)

4.28.4.82 tuple_ids* new_tuple_ids (int * tuple)

4.28.4.83 htable* prepare_pipelined_join (htable ** htap, htable ** htbp, htable * joined_table, int no_destroy)

4.28.4.84 void print_ht (htable * ht, int short_name)

4.28.4.85 void print_ht_h (htable * ht, int short_name)

4.28.4.86 void print_ht_line (htable * ht)

4.28.4.87 void print_ht_mysql (htable * ht)

4.28.4.88 htable* project_distinct_table (htable * ht, llist * cols, int distinct)

4.28.4.89 htable* project_join_tables (htable * hta, htable * htb)

4.28.4.90 htable* project_table (htable * ht, llist * cols)

4.28.4.91 htable* purge_htable (htable * ht)

4.28.4.92 void rebuild_histogram (htable * table)

4.28.4.93 void replace_data (htable * data, htable * ht)

4.28.4.94 void resetset_variable_hashtables (llist * htables)

4.28.4.95 void set_it_object (void ∗ *a,* int *i,* int *c,* void ∗ *obj*)

4.28.4.96 void set_tuple_value (double *value,* int *row,* htable ∗ *ht*)

4.28.4.97 void set_value_sort (void ∗ *vht,* int *row,* int *column,* int *value*)

4.28.4.98 void set_variable_hashtable (htable ∗ *ht*)

4.28.4.99 void set_variable_hashtables (llist ∗ *htables*)

4.28.4.100 void sort_ht_iterator (htable ∗ *ht*)

4.28.4.101 void track_alive (htable ∗ *ht*)

4.28.4.102 void un_delete_tuple (htable ∗ *ht,* int *tuple*)

4.28.4.103 void un_delete_tuple_old (htable ∗ *ht,* int *tuple*)

4.28.4.104 void update_alive (alive_table ∗ *at,* htable ∗ *ht*)

4.28.4.105 void update_current_domains (htable ∗ *ht,* light_stack ∗ *undo_value_stack*)

4.28.4.106 void update_histogram (htable ∗ *table*)

4.29 hashtable_search.c File Reference

```
#include "hashtable.h"
#include "hashtable_search.h"
#include "utils.h"
#include "active_tuple.h"
#include "rs2.h"
#include "decomposition.h"
#include "splitter.h"
#include "llist.h"
#include "sort.h"
```

Functions

- htable ∗ get_projected_on_variables (htable ∗ht, llist ∗variablelist)
- htable ∗ join_filter_relations_with_all_sols (llist ∗htablelist, int nojoin, int nofilter, char ∗optionalname)
- htable ∗ project_with_tree (htable ∗ht, llist ∗varlist)
- htable ∗ project_with_tree_payload (htable ∗ht, llist ∗varlist, tree_map ∗∗tree_only, void ∗get_payload(int param), int param)
- void sort_with_tree (htable ∗ht, llist ∗varlist, int ∗vmap)
- llist ∗ project_combine_relations (llist ∗htables, llist ∗varlist, int filter)
- llist ∗ generate_tables_for_scopes_using_relations (llist ∗list_of_lvids, llist ∗using_rels, int id, int scopes_int-_p, int record_stats)
- llist ∗ generate_tables_for_scopes_using_relations_noproject (llist ∗list_of_lvids, llist ∗using_rels, int id, int scopes_int_p, int record_stats)
- llist ∗ join_project_multiple_in_multiple_out (llist ∗htablelist, llist ∗list_of_domain_lists, int nofilteroriginal, int id)
- int filter_this_relation_with_1_sol (htable ∗htfilter, llist ∗given_htablelist, set ∗undo_set, light_stack ∗undo_-stack)
- int filter_relations_with_1_sol (llist ∗htablelist)

- llist ∗ join_project_multiple_in_multiple_out_noextend (llist ∗htablelist, llist ∗list_of_domain_lists, int id)
- llist ∗ join_project_multiple_in_multiple_out_noextend_given_tables (llist ∗first_tables, llist ∗htablelist_ofrest, llist ∗list_of_domain_lists, int id)
- int process_with_all_sols (llist ∗htablelist, active_tuple ∗ac, options catgegory)
- int process_with_all_sols_first_var (llist ∗htablelist, active_tuple ∗ac, int first_v, options category)

4.29.1 Function Documentation

4.29.1.1 int filter_relations_with_1_sol (**llist** ∗ *htablelist*)

4.29.1.2 int filter_this_relation_with_1_sol (**htable** ∗ *htfilter,* **llist** ∗ *given_htablelist,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack*)

4.29.1.3 **llist**∗ generate_tables_for_scopes_using_relations (**llist** ∗ *list_of_lvids,* **llist** ∗ *using_rels,* int *id,* int *scopes_int_p,* int *record_stats*)

4.29.1.4 **llist**∗ generate_tables_for_scopes_using_relations_noproject (**llist** ∗ *list_of_lvids,* **llist** ∗ *using_rels,* int *id,* int *scopes_int_p,* int *record_stats*)

4.29.1.5 **htable**∗ get_projected_on_variables (**htable** ∗ *ht,* **llist** ∗ *variablelist*)

4.29.1.6 **htable**∗ join_filter_relations_with_all_sols (**llist** ∗ *htablelist,* int *nojoin,* int *nofilter,* char ∗ *optionalname*)

4.29.1.7 **llist**∗ join_project_multiple_in_multiple_out (**llist** ∗ *htablelist,* **llist** ∗ *list_of_domain_lists,* int *nofilteroriginal,* int *id*)

4.29.1.8 **llist**∗ join_project_multiple_in_multiple_out_noextend (**llist** ∗ *htablelist,* **llist** ∗ *list_of_domain_lists,* int *id*)

4.29.1.9 **llist**∗ join_project_multiple_in_multiple_out_noextend_given_tables (**llist** ∗ *first_tables,* **llist** ∗ *htablelist_ofrest,* **llist** ∗ *list_of_domain_lists,* int *id*)

4.29.1.10 int process_with_all_sols (**llist** ∗ *htablelist,* **active_tuple** ∗ *ac,* **options** *catgegory*)

4.29.1.11 int process_with_all_sols_first_var (**llist** ∗ *htablelist,* **active_tuple** ∗ *ac,* int *first_v,* **options** *category*)

4.29.1.12 **llist**∗ project_combine_relations (**llist** ∗ *htables,* **llist** ∗ *varlist,* int *filter*)

4.29.1.13 **htable**∗ project_with_tree (**htable** ∗ *ht,* **llist** ∗ *varlist*)

4.29.1.14 **htable**∗ project_with_tree_payload (**htable** ∗ *ht,* **llist** ∗ *varlist,* **tree_map** ∗∗ *tree_only,* void ∗ *get_payloadint param,* int *param*)

4.29.1.15 void sort_with_tree (**htable** ∗ *ht,* **llist** ∗ *varlist,* int ∗ *vmap*)

4.30 hashtable_search.h File Reference

```
#include "utils.h"
#include "hashtable.h"
#include "llist.h"
#include "active_tuple.h"
```

Functions

- htable ∗ get_projected_on_variables (htable ∗ht, llist ∗sepnames)

- htable ∗ join_filter_relations_with_all_sols (llist ∗htablelist, int nojoin, int nofliter, char ∗optionalname)
- htable ∗ project_with_tree (htable ∗ht, llist ∗varlist)
- htable ∗ project_with_tree_payload (htable ∗ht, llist ∗varlist, tree_map ∗∗tree_only, void ∗get_payload(int param), int param)
- llist ∗ project_combine_relations (llist ∗htables, llist ∗variablelist, int filter)
- llist ∗ generate_tables_for_scopes_using_relations (llist ∗list_of_lvids, llist ∗using_rels, int id, int scopes_int_p, int record_stats)
- llist ∗ generate_tables_for_scopes_using_relations_noproject (llist ∗list_of_lvids, llist ∗using_rels, int id, int scopes_int_p, int record_stats)
- int filter_relations_with_1_sol (llist ∗htablelist)
- int filter_this_relation_with_1_sol (htable ∗htfilter, llist ∗given_htablelist, set ∗undo_set, light_stack ∗undo_stack)
- llist ∗ join_project_multiple_in_multiple_out (llist ∗htablelist, llist ∗list_of_domain_lists, int nofilteroriginal, int id)
- void sort_with_tree (htable ∗ht, llist ∗varlist, int ∗vmap)
- void find_all_supports_with_all_solutions_with_fc_noextend (llist ∗htablelist, llist ∗vars)
- llist ∗ join_project_multiple_in_multiple_out_noextend (llist ∗htablelist, llist ∗list_of_domain_lists, int id)
- llist ∗ join_project_multiple_in_multiple_out_noextend_given_tables (llist ∗first_tables, llist ∗htablelist_ofrest, llist ∗list_of_domain_lists, int id)
- int process_with_all_sols (llist ∗htablelist, active_tuple ∗ac, options catgegory)
- int process_with_all_sols_first_var (llist ∗htablelist, active_tuple ∗ac, int first_v, options catgegory)

4.30.1 Detailed Description

Provides tree-based algorithms for processing relations.

4.30.2 Function Documentation

4.30.2.1 int filter_relations_with_1_sol (llist ∗ *htablelist*)

4.30.2.2 int filter_this_relation_with_1_sol (htable ∗ *htfilter*, llist ∗ *given_htablelist*, set ∗ *undo_set*, light_stack ∗ *undo_stack*)

4.30.2.3 void find_all_supports_with_all_solutions_with_fc_noextend (llist ∗ *htablelist*, llist ∗ *vars*)

4.30.2.4 llist∗ generate_tables_for_scopes_using_relations (llist ∗ *list_of_lvids*, llist ∗ *using_rels*, int *id*, int *scopes_int_p*, int *record_stats*)

4.30.2.5 llist∗ generate_tables_for_scopes_using_relations_noproject (llist ∗ *list_of_lvids*, llist ∗ *using_rels*, int *id*, int *scopes_int_p*, int *record_stats*)

4.30.2.6 htable∗ get_projected_on_variables (htable ∗ *ht*, llist ∗ *sepnames*)

4.30.2.7 htable∗ join_filter_relations_with_all_sols (llist ∗ *htablelist*, int *nojoin*, int *nofliter*, char ∗ *optionalname*)

4.30.2.8 llist∗ join_project_multiple_in_multiple_out (llist ∗ *htablelist*, llist ∗ *list_of_domain_lists*, int *nofilteroriginal*, int *id*)

4.30.2.9 llist∗ join_project_multiple_in_multiple_out_noextend (llist ∗ *htablelist*, llist ∗ *list_of_domain_lists*, int *id*)

4.30.2.10 llist∗ join_project_multiple_in_multiple_out_noextend_given_tables (llist ∗ *first_tables*, llist ∗ *htablelist_ofrest*, llist ∗ *list_of_domain_lists*, int *id*)

4.30.2.11 int process_with_all_sols (llist ∗ *htablelist*, active_tuple ∗ *ac*, options *catgegory*)

4.30.2.12 int process_with_all_sols_first_var (**llist** ∗ *htablelist,* **active_tuple** ∗ *ac,* int *first_v,* **options** *catgegory*)

4.30.2.13 **llist**∗ project_combine_relations (**llist** ∗ *htables,* **llist** ∗ *variablelist,* int *filter*)

4.30.2.14 **htable**∗ project_with_tree (**htable** ∗ *ht,* **llist** ∗ *varlist*)

4.30.2.15 **htable**∗ project_with_tree_payload (**htable** ∗ *ht,* **llist** ∗ *varlist,* **tree_map** ∗∗ *tree_only,* void ∗ *get_payloadint param,* int *param*)

4.30.2.16 void sort_with_tree (**htable** ∗ *ht,* **llist** ∗ *varlist,* int ∗ *vmap*)

4.31 heap.c File Reference

```
#include <stdio.h>
#include "heap.h"
#include "utils.h"
```

Functions

- heap ∗ new_heap (int size)
- void destroy_heap (heap ∗h)
- int heap_parent (int i)
- int heap_left (int i)
- int heap_right (int i)
- void heap_exchange (int i, int j, heap ∗h, void set_heap_position(int, void ∗))
- void min_heapify (heap ∗h, int i, int get_heap_value(void ∗), void set_heap_position(int, void ∗))
- void build_min_heap (heap ∗h, int get_heap_value(void ∗), void set_heap_position(int, void ∗))
- void ∗ heap_min (heap ∗h)
- void ∗ heap_extract_min (heap ∗h, int get_heap_value(void ∗), void set_heap_position(int, void ∗))
- void heap_move_up (heap ∗h, int i, int get_heap_value(void ∗), void set_heap_position(int, void ∗))
- void heap_update (heap ∗h, int i, int val, int get_heap_value(void ∗), void set_heap_value(int, void ∗), void set_heap_position(int, void ∗))
- void heap_insert (heap ∗h, void ∗item, int get_heap_value(void ∗), void set_heap_value(int, void ∗), void set_heap_position(int, void ∗))
- void heap_delete (heap ∗h, int i, int get_heap_value(void ∗), void set_heap_value(int, void ∗), void set_heap_position(int, void ∗))
- void print_heap (heap ∗h, int get_heap_value(void ∗))

4.31.1 Function Documentation

4.31.1.1 void build_min_heap (**heap** ∗ *h,* int *get_heap_valuevoid* ∗*,* void *set_heap_positionint, void* ∗)

4.31.1.2 void destroy_heap (**heap** ∗ *h*)

4.31.1.3 void heap_delete (**heap** ∗ *h,* int *i,* int *get_heap_valuevoid* ∗*,* void *set_heap_valueint, void* ∗*,* void *set_heap_positionint, void* ∗)

4.31.1.4 void heap_exchange (int *i,* int *j,* **heap** ∗ *h,* void *set_heap_positionint, void* ∗)

4.31.1.5 void∗ heap_extract_min (**heap** ∗ *h,* int *get_heap_valuevoid* ∗*,* void *set_heap_positionint, void* ∗)

4.31.1.6 void heap_insert (**heap** ∗ *h,* void ∗ *item,* int *get_heap_valuevoid* ∗*,* void *set_heap_valueint, void* ∗*,* void *set_heap_positionint, void* ∗)

4.31.1.7 int heap_left (int *i*)

4.31.1.8 void∗ heap_min (**heap** ∗ *h*)

4.31.1.9 void heap_move_up (**heap** ∗ *h,* int *i,* int *get_heap_valuevoid* ∗*,* void *set_heap_positionint, void* ∗)

4.31.1.10 int heap_parent (int *i*)

4.31.1.11 int heap_right (int *i*)

4.31.1.12 void heap_update (**heap** ∗ *h,* int *i,* int *val,* int *get_heap_valuevoid* ∗*,* void *set_heap_valueint, void* ∗*,* void *set_heap_positionint, void* ∗)

4.31.1.13 void min_heapify (**heap** ∗ *h,* int *i,* int *get_heap_valuevoid* ∗*,* void *set_heap_positionint, void* ∗)

4.31.1.14 **heap**∗ new_heap (int *size*)

4.31.1.15 void print_heap (**heap** ∗ *h,* int *get_heap_valuevoid* ∗)

4.32 heap.h File Reference

Classes

- struct heap

Macros

- #define INF 2147483646

Typedefs

- typedef struct heap heap

Functions

- heap ∗ new_heap (int size)
- void destroy_heap (heap ∗h)
- void heap_update (heap ∗h, int i, int val, int get_heap_value(void ∗), void set_heap_value(int, void ∗), void set_heap_position(int, void ∗))
- int heap_parent (int i)
- int heap_left (int i)
- int heap_right (int i)
- void heap_exchange (int i, int j, heap ∗h, void set_heap_position(int, void ∗))
- void min_heapify (heap ∗h, int i, int get_heap_value(void ∗), void set_heap_position(int, void ∗))
- void build_min_heap (heap ∗h, int get_heap_value(void ∗), void set_heap_position(int, void ∗))
- void ∗ heap_min (heap ∗h)
- void ∗ heap_extract_min (heap ∗h, int get_heap_value(void ∗), void set_heap_position(int, void ∗))
- void heap_move_up (heap ∗h, int i, int get_heap_value(void ∗), void set_heap_position(int, void ∗))
- void heap_insert (heap ∗h, void ∗item, int get_heap_value(void ∗), void set_heap_value(int, void ∗), void set_heap_position(int, void ∗))
- void heap_delete (heap ∗h, int i, int get_heap_value(void ∗), void set_heap_value(int, void ∗), void set_heap_position(int, void ∗))
- void print_heap (heap ∗hd, int get_heap_value(void ∗))

4.32.1 Detailed Description

Provides a heap implementation.

4.32.2 Macro Definition Documentation

4.32.2.1 #define INF 2147483646

4.32.3 Typedef Documentation

4.32.3.1 typedef struct heap heap

4.32.4 Function Documentation

4.32.4.1 void build_min_heap (heap ∗ *h*, int *get_heap_valuevoid* ∗, void *set_heap_positionint, void* ∗)

4.32.4.2 void destroy_heap (heap ∗ *h*)

4.32.4.3 void heap_delete (heap ∗ *h*, int *i*, int *get_heap_valuevoid* ∗, void *set_heap_valueint, void* ∗, void *set_heap_positionint, void* ∗)

4.32.4.4 void heap_exchange (int *i*, int *j*, heap ∗ *h*, void *set_heap_positionint, void* ∗)

4.32.4.5 void∗ heap_extract_min (heap ∗ *h*, int *get_heap_valuevoid* ∗, void *set_heap_positionint, void* ∗)

4.32.4.6 void heap_insert (heap ∗ *h*, void ∗ *item*, int *get_heap_valuevoid* ∗, void *set_heap_valueint, void* ∗, void *set_heap_positionint, void* ∗)

4.32.4.7 int heap_left (int *i*)

4.32.4.8 void∗ heap_min (heap ∗ *h*)

4.32.4.9 void heap_move_up (heap ∗ *h*, int *i*, int *get_heap_valuevoid* ∗, void *set_heap_positionint, void* ∗)

4.32.4.10 int heap_parent (int *i*)

4.32.4.11 int heap_right (int *i*)

4.32.4.12 void heap_update (heap ∗ *h*, int *i*, int *val*, int *get_heap_valuevoid* ∗, void *set_heap_valueint, void* ∗, void *set_heap_positionint, void* ∗)

4.32.4.13 void min_heapify (heap ∗ *h*, int *i*, int *get_heap_valuevoid* ∗, void *set_heap_positionint, void* ∗)

4.32.4.14 heap∗ new_heap (int *size*)

4.32.4.15 void print_heap (heap ∗ *hd*, int *get_heap_valuevoid* ∗)

4.33 histogram.c File Reference

```
#include "histogram.h"
#include "utils.h"
```

4.34 histogram.h File Reference

4.34.1 Detailed Description

Provides histogram implementation for the relations.

4.35 llist.c File Reference

```
#include "llist.h"
#include "domains.h"
#include "utils.h"
```

Functions

- llist_node ∗ new_lnode (void)
- llist ∗ new_llist (void)
- void destroy_list_nodes_only (llist ∗list)
- void destroy_list_nodes (llist_node ∗nodes)
- void destroy_list_nodes_body (llist_node ∗nodes)
- void destroy_list (llist ∗list)
- void destroy_list_body (llist ∗list)
- void add_int (int ibody, llist ∗list)
- void add_node (void ∗body, llist ∗list)
- void add_node_tail (void ∗body, llist ∗list)
- void add_node_tail_distinct (void ∗body, llist ∗list)
- void add_int_tail_distinct (int ibody, llist ∗list)
- void add_int_tail (int ibody, llist ∗list)
- void add_this_node_tail (llist_node ∗node, llist ∗list)
- void add_this_node (llist_node ∗node, llist ∗list)
- llist_node ∗ find_remove_node_of_body (void ∗value, int compare(void ∗value, void ∗bodyb), llist ∗list)
- int restore_element_to_same_position (llist_node ∗node, llist ∗list)
- llist_node ∗ remove_node_no_check_no_null (llist_node ∗node, llist ∗list)
- llist_node ∗ remove_node_no_check (llist_node ∗node, llist ∗list)
- void move_to_tail (llist_node ∗node, llist ∗list)
- void move_to_head (llist_node ∗node, llist ∗list)
- llist_node ∗ remove_node (llist_node ∗node, llist ∗list)
- void remove_node_of_body (void ∗body, llist ∗list)
- llist_node ∗ remove_single_node_of_body (void ∗body, llist ∗list)
- llist_node ∗ find_node_of_value (void ∗value, int compare(void ∗value, void ∗bodyb), llist ∗list)
- llist_node ∗ find_node_of_ivalue (int value, llist ∗list)
- llist_node ∗ get_node_of_body (void ∗body, llist ∗list)
- llist_node ∗ remove_head (llist ∗list)
- llist_node ∗ remove_tail (llist ∗list)
- void move_llist_to (llist ∗list, llist ∗destination_list)
- void copy_llist_to (llist ∗list, llist ∗destination_list)
- void copy_llist_to_distinct (llist ∗list, llist ∗destination_list)
- llist ∗ copy_int_llist_add_pos (llist ∗list, int pos0)
- llist ∗ copy_llist (llist ∗list)
- llist ∗ copy_value_llist (llist ∗list)
- int is_empty (llist ∗list)
- int max_int (llist ∗list, int get_value(void ∗body))
- int min_int (llist ∗list, int get_value(void ∗body))

- llist_node ∗ remove_max (llist ∗list, int get_value(void ∗body))
- llist_node ∗ remove_min (llist ∗list, int get_value(void ∗body))
- llist_node ∗ get_max (llist ∗list, int get_value(void ∗body))
- llist_node ∗ get_min (llist ∗list, int get_value(void ∗body))
- llist_node ∗ remove_node_of_ibody (int ibody, llist ∗list)
- llist_node ∗ get_node_of_ibody (int ibody, llist ∗list)
- light_stack ∗ new_light_stack (void)
- light_stack_node ∗ new_light_stack_node (int value)
- void push_to_light_stack (int value, void ∗body, light_stack ∗lis)
- int pop_light_stack (light_stack ∗lis, void ∗∗body_holder)
- void push_s (void ∗item, llist ∗stack)
- void ∗ pop_s (llist ∗stack)
- void ∗ peek_s (llist ∗stack)
- void ∗ llist_dequeue (llist ∗queue)
- void llist_enqueue (void ∗object, llist ∗queue)

4.35.1 Function Documentation

4.35.1.1 void add_int (int *ibody,* llist ∗ *list*)

4.35.1.2 void add_int_tail (int *ibody,* llist ∗ *list*)

4.35.1.3 void add_int_tail_distinct (int *ibody,* llist ∗ *list*)

4.35.1.4 void add_node (void ∗ *body,* llist ∗ *list*)

4.35.1.5 void add_node_tail (void ∗ *body,* llist ∗ *list*)

4.35.1.6 void add_node_tail_distinct (void ∗ *body,* llist ∗ *list*)

4.35.1.7 void add_this_node (llist_node ∗ *node,* llist ∗ *list*)

4.35.1.8 void add_this_node_tail (llist_node ∗ *node,* llist ∗ *list*)

4.35.1.9 llist∗ copy_int_llist_add_pos (llist ∗ *list,* int *pos0*)

4.35.1.10 llist∗ copy_llist (llist ∗ *list*)

4.35.1.11 void copy_llist_to (llist ∗ *list,* llist ∗ *destination_list*)

4.35.1.12 void copy_llist_to_distinct (llist ∗ *list,* llist ∗ *destination_list*)

4.35.1.13 llist∗ copy_value_llist (llist ∗ *list*)

4.35.1.14 void destroy_list (llist ∗ *list*)

4.35.1.15 void destroy_list_body (llist ∗ *list*)

4.35.1.16 void destroy_list_nodes (llist_node ∗ *nodes*)

4.35.1.17 void destroy_list_nodes_body (llist_node ∗ *nodes*)

4.35.1.18 void destroy_list_nodes_only (llist ∗ *list*)

4.35.1.19 llist_node∗ find_node_of_ivalue (int *value,* llist ∗ *list*)

4.35.1.20 llist_node∗ find_node_of_value (void ∗ *value,* int *comparevoid ∗value, void ∗bodyb,* llist ∗ *list*)

4.35.1.21 llist_node∗ find_remove_node_of_body (void ∗ *value,* int *comparevoid ∗value, void ∗bodyb,* llist ∗ *list*)

4.35.1.22 llist_node∗ get_max (llist ∗ *list,* int *get_valuevoid ∗body*)

4.35.1.23 llist_node∗ get_min (llist ∗ *list,* int *get_valuevoid ∗body*)

4.35.1.24 llist_node∗ get_node_of_body (void ∗ *body,* llist ∗ *list*)

4.35.1.25 llist_node∗ get_node_of_ibody (int *ibody,* llist ∗ *list*)

4.35.1.26 int is_empty (llist ∗ *list*)

4.35.1.27 void∗ llist_dequeue (llist ∗ *queue*)

4.35.1.28 void llist_enqueue (void ∗ *object,* llist ∗ *queue*)

4.35.1.29 int max_int (llist ∗ *list,* int *get_valuevoid ∗body*)

4.35.1.30 int min_int (llist ∗ *list,* int *get_valuevoid ∗body*)

4.35.1.31 void move_llist_to (llist ∗ *list,* llist ∗ *destination_list*)

4.35.1.32 void move_to_head (llist_node ∗ *node,* llist ∗ *list*)

4.35.1.33 void move_to_tail (llist_node ∗ *node,* llist ∗ *list*)

4.35.1.34 light_stack∗ new_light_stack (void)

4.35.1.35 light_stack_node∗ new_light_stack_node (int *value*)

4.35.1.36 llist∗ new_llist (void)

4.35.1.37 llist_node∗ new_lnode (void)

4.35.1.38 void∗ peek_s (llist ∗ *stack*)

4.35.1.39 int pop_light_stack (light_stack ∗ *lis,* void ∗∗ *body_holder*)

4.35.1.40 void∗ pop_s (llist ∗ *stack*)

4.35.1.41 void push_s (void ∗ *item,* llist ∗ *stack*)

4.35.1.42 void push_to_light_stack (int *value,* void ∗ *body,* light_stack ∗ *lis*)

4.35.1.43 llist_node∗ remove_head (llist ∗ *list*)

4.35.1.44 llist_node∗ remove_max (llist ∗ *list,* int *get_valuevoid ∗body*)

4.35.1.45 llist_node∗ remove_min (llist ∗ *list,* int *get_valuevoid ∗body*)

4.35.1.46 llist_node∗ remove_node (llist_node ∗ *node,* llist ∗ *list*)

4.35.1.47 llist_node∗ remove_node_no_check (llist_node ∗ *node,* llist ∗ *list*)

4.35.1.48 llist_node∗ remove_node_no_check_no_null (llist_node ∗ *node,* llist ∗ *list*)

4.35.1.49 void remove_node_of_body (void ∗ *body,* llist ∗ *list*)

4.35.1.50 llist_node∗ remove_node_of_ibody (int *ibody,* llist ∗ *list*)

4.35.1.51 llist_node∗ remove_single_node_of_body (void ∗ *body,* llist ∗ *list*)

4.35.1.52 llist_node∗ remove_tail (llist ∗ *list*)

4.35.1.53 int restore_element_to_same_position (llist_node ∗ *node,* llist ∗ *list*)

4.36 llist.h File Reference

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

Classes

- struct llist_node
- struct llist
- struct light_stack
- struct light_stack_node

Typedefs

- typedef struct llist_node llist_node
- typedef struct llist llist
- typedef struct light_stack light_stack
- typedef struct light_stack_node light_stack_node

Functions

- llist_node ∗ new_lnode (void)
- llist ∗ new_llist (void)
- void add_int (int ibody, llist ∗list)
- void add_node (void ∗body, llist ∗list)
- void add_node_tail (void ∗body, llist ∗list)
- void add_node_tail_distinct (void ∗body, llist ∗list)
- void add_int_tail (int ibody, llist ∗list)
- void add_int_tail_distinct (int ibody, llist ∗list)
- void add_this_node_tail (llist_node ∗node, llist ∗list)
- void add_this_node (llist_node ∗node, llist ∗list)
- llist_node ∗ remove_node (llist_node ∗node, llist ∗list)
- int restore_element_to_same_position (llist_node ∗node, llist ∗list)
- llist_node ∗ remove_node_no_check_no_null (llist_node ∗node, llist ∗list)
- llist_node ∗ remove_node_no_check (llist_node ∗node, llist ∗list)
- void remove_node_of_body (void ∗body, llist ∗list)
- llist_node ∗ find_remove_node_of_body (void ∗value, int compare(void ∗value, void ∗bodyb), llist ∗list)
- llist_node ∗ get_node_of_body (void ∗body, llist ∗list)
- llist_node ∗ find_node_of_value (void ∗value, int compare(void ∗value, void ∗bodyb), llist ∗list)
- llist_node ∗ find_node_of_ivalue (int value, llist ∗list)
- llist_node ∗ remove_single_node_of_body (void ∗body, llist ∗list)

- llist_node ∗ remove_head (llist ∗list)
- llist_node ∗ remove_tail (llist ∗list)
- llist ∗ copy_value_llist (llist ∗list)
- llist ∗ copy_llist (llist ∗list)
- void copy_llist_to (llist ∗list, llist ∗destination_list)
- void copy_llist_to_distinct (llist ∗list, llist ∗destination_list)
- llist ∗ copy_int_llist_add_pos (llist ∗list, int pos0)
- int is_empty (llist ∗list)
- void destroy_list_nodes_only (llist ∗list)
- void destroy_list_nodes (llist_node ∗llist_node)
- void destroy_list_nodes_body (llist_node ∗llist_node)
- void destroy_list (llist ∗list)
- void destroy_list_body (llist ∗list)
- int max_int (llist ∗list, int get_value(void ∗body))
- llist_node ∗ remove_max (llist ∗list, int get_value(void ∗body))
- llist_node ∗ remove_min (llist ∗list, int get_value(void ∗body))
- llist_node ∗ get_max (llist ∗list, int get_value(void ∗body))
- llist_node ∗ get_min (llist ∗list, int get_value(void ∗body))
- void move_to_tail (llist_node ∗node, llist ∗list)
- void move_to_head (llist_node ∗node, llist ∗list)
- llist_node ∗ remove_node_of_ibody (int ibody, llist ∗list)
- llist_node ∗ get_node_of_ibody (int ibody, llist ∗list)
- int min_int (llist ∗list, int get_value(void ∗body))
- void move_llist_to (llist ∗list, llist ∗destination_list)
- light_stack ∗ new_light_stack (void)
- light_stack_node ∗ new_light_stack_node (int value)
- void push_to_light_stack (int value, void ∗body, light_stack ∗lis)
- int pop_light_stack (light_stack ∗lis, void ∗∗body_holder)
- void push_s (void ∗item, llist ∗stack)
- void ∗ pop_s (llist ∗stack)
- void ∗ peek_s (llist ∗stack)
- void ∗ llist_dequeue (llist ∗queue)
- void llist_enqueue (void ∗object, llist ∗queue)

4.36.1 Detailed Description

Provides linked-list implementation.

4.36.2 Typedef Documentation

4.36.2.1 typedef struct **light_stack light_stack**

4.36.2.2 typedef struct **light_stack_node light_stack_node**

4.36.2.3 typedef struct **llist llist**

4.36.2.4 typedef struct **llist_node llist_node**

4.36.3 Function Documentation

4.36.3.1 void add_int (int *ibody,* **llist** ∗ *list*)

4.36.3.2 void add_int_tail (int *ibody,* **llist** ∗ *list*)

4.36.3.3 void add_int_tail_distinct (int *ibody,* llist ∗ *list*)

4.36.3.4 void add_node (void ∗ *body,* llist ∗ *list*)

4.36.3.5 void add_node_tail (void ∗ *body,* llist ∗ *list*)

4.36.3.6 void add_node_tail_distinct (void ∗ *body,* llist ∗ *list*)

4.36.3.7 void add_this_node (llist_node ∗ *node,* llist ∗ *list*)

4.36.3.8 void add_this_node_tail (llist_node ∗ *node,* llist ∗ *list*)

4.36.3.9 llist∗ copy_int_llist_add_pos (llist ∗ *list,* int *pos0*)

4.36.3.10 llist∗ copy_llist (llist ∗ *list*)

4.36.3.11 void copy_llist_to (llist ∗ *list,* llist ∗ *destination_list*)

4.36.3.12 void copy_llist_to_distinct (llist ∗ *list,* llist ∗ *destination_list*)

4.36.3.13 llist∗ copy_value_llist (llist ∗ *list*)

4.36.3.14 void destroy_list (llist ∗ *list*)

4.36.3.15 void destroy_list_body (llist ∗ *list*)

4.36.3.16 void destroy_list_nodes (llist_node ∗ *llist_node*)

4.36.3.17 void destroy_list_nodes_body (llist_node ∗ *llist_node*)

4.36.3.18 void destroy_list_nodes_only (llist ∗ *list*)

4.36.3.19 llist_node∗ find_node_of_ivalue (int *value,* llist ∗ *list*)

4.36.3.20 llist_node∗ find_node_of_value (void ∗ *value,* int *comparevoid ∗value, void ∗bodyb,* llist ∗ *list*)

4.36.3.21 llist_node∗ find_remove_node_of_body (void ∗ *value,* int *comparevoid ∗value, void ∗bodyb,* llist ∗ *list*)

4.36.3.22 llist_node∗ get_max (llist ∗ *list,* int *get_valuevoid ∗body*)

4.36.3.23 llist_node∗ get_min (llist ∗ *list,* int *get_valuevoid ∗body*)

4.36.3.24 llist_node∗ get_node_of_body (void ∗ *body,* llist ∗ *list*)

4.36.3.25 llist_node∗ get_node_of_ibody (int *ibody,* llist ∗ *list*)

4.36.3.26 int is_empty (llist ∗ *list*)

4.36.3.27 void∗ llist_dequeue (llist ∗ *queue*)

4.36.3.28 void llist_enqueue (void ∗ *object,* llist ∗ *queue*)

4.36.3.29 int max_int (llist ∗ *list,* int *get_valuevoid ∗body*)

4.36.3.30 int min_int (llist ∗ *list,* int *get_valuevoid ∗body*)

4.36.3.31 void move_llist_to (llist ∗ *list,* llist ∗ *destination_list*)

4.36.3.32 void move_to_head (llist_node ∗ *node,* llist ∗ *list*)

4.36.3.33 void move_to_tail (llist_node ∗ *node,* llist ∗ *list*)

4.36.3.34 light_stack∗ new_light_stack (void)

4.36.3.35 light_stack_node∗ new_light_stack_node (int *value*)

4.36.3.36 llist∗ new_llist (void)

4.36.3.37 llist_node∗ new_lnode (void)

4.36.3.38 void∗ peek_s (llist ∗ *stack*)

4.36.3.39 int pop_light_stack (light_stack ∗ *lis,* void ∗∗ *body_holder*)

4.36.3.40 void∗ pop_s (llist ∗ *stack*)

4.36.3.41 void push_s (void ∗ *item,* llist ∗ *stack*)

4.36.3.42 void push_to_light_stack (int *value,* void ∗ *body,* light_stack ∗ *lis*)

4.36.3.43 llist_node∗ remove_head (llist ∗ *list*)

4.36.3.44 llist_node∗ remove_max (llist ∗ *list,* int *get_valuevoid* ∗*body*)

4.36.3.45 llist_node∗ remove_min (llist ∗ *list,* int *get_valuevoid* ∗*body*)

4.36.3.46 llist_node∗ remove_node (llist_node ∗ *node,* llist ∗ *list*)

4.36.3.47 llist_node∗ remove_node_no_check (llist_node ∗ *node,* llist ∗ *list*)

4.36.3.48 llist_node∗ remove_node_no_check_no_null (llist_node ∗ *node,* llist ∗ *list*)

4.36.3.49 void remove_node_of_body (void ∗ *body,* llist ∗ *list*)

4.36.3.50 llist_node∗ remove_node_of_ibody (int *ibody,* llist ∗ *list*)

4.36.3.51 llist_node∗ remove_single_node_of_body (void ∗ *body,* llist ∗ *list*)

4.36.3.52 llist_node∗ remove_tail (llist ∗ *list*)

4.36.3.53 int restore_element_to_same_position (llist_node ∗ *node,* llist ∗ *list*)

4.37 main.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
#include <string.h>
#include <math.h>
#include <getopt.h>
#include "parser/C_CSPParserCallback.h"
#include "globals.h"
#include "procedures.h"
#include "utils.h"
#include "ac.h"
#include "randomize.h"
#include "query_graph.h"
#include "sac.h"
#include "rpwc.h"
#include "terminal.h"
#include "do_db.h"
#include "parsers.h"
#include "normalize_constraints.h"
#include "decomposition.h"
#include "oneall.h"
#include "solution.h"
#include <signal.h>
```

Functions

- void sigxcpu_handler (int signo)
- int main (int argc, char ∗∗argv)
- void check_ok2 (variables ∗vars)
- void check_ok (main_structure ∗m_s)

Variables

- main_structure ∗ m_s_g

4.37.1 Detailed Description

Provides the code for parsing the input parameters and runs the program.

Finding One solution:

(for pre-processing only, remove '-Sx')

GAC :: -f [file] -k -uDD -DGA -F -Sx -v-1
maxRPWC :: -f [file] -k -uDD -DRP -F -Sx -v-1

Random: cl+proj-wR(∗,2)C :: -f [file] -k -Tmfrstonenosp -xpb -qms -Drr -s2 -b2s -uDD -F -Sx
Random: cl+proj-wR(∗,3)C :: -f [file] -k -Tmfrstonenosp -xpb -qms -Drr -s3 -b2s -uDD -F -Sx
Random: cl+proj-wR(∗,4)C :: -f [file] -k -Tmfrstonenosp -xpb -qms -Drr -s4 -b2s -uDD -F -Sx
Random: cl+proj-wR(∗,|C|)C :: -f [file] -k -Tmfrstonenosp -xpb -qms -s0 -b2s -uDD -F -Sx

```
Random: cl-wR(∗,2)C :: -f [file] -k -Tmfrstonenosp -xss -qms -Drr -s2 -b2s -uDD -F -Sx
Random: cl-wR(∗,3)C :: -f [file] -k -Tmfrstonenosp -xss -qms -Drr -s3 -b2s -uDD -F -Sx
Random: cl-wR(∗,4)C :: -f [file] -k -Tmfrstonenosp -xss -qms -Drr -s4 -b2s -uDD -F -Sx
Random: cl-wR(∗,|C|)C :: -f [file] -k -Tmfrstonenosp -xss -qms -s0 -b2s -uDD -F -Sx


wR(∗,2)C :: -f [file] -k -Tmfrstonenosp -xpb -Drr -s2 -b1 -uDD -F -Sx -v-1
wR(∗,3)C :: -f [file] -k -Tmfrstonenosp -xpb -Drr -s3 -b1 -uDD -F -Sx -v-1
wR(∗,4)C :: -f [file] -k -Tmfrstonenosp -xpb -Drr -s4 -b1 -uDD -F -Sx -v-1


Static: cl-wR(∗,2)C :: -f [file] -k -Tmfrstonenosp -xss -qmr -Drr -s2 -b2n0 -uDD -F -Sx
Static: cl-wR(∗,3)C :: -f [file] -k -Tmfrstonenosp -xss -qmr -Drr -s3 -b2n0 -uDD -F -Sx
Static: cl-wR(∗,4)C :: -f [file] -k -Tmfrstonenosp -xss -qmr -Drr -s4 -b2n0 -uDD -F -Sx
Static: cl-wR(∗,|C|)C :: -f [file] -k -Tmfrstonenosp -xss -qmr -s0 -b2n0 -uDD -F -Sx


Static: cl+proj-wR(∗,2)C :: -f [file] -k -Tmfrstonenosp -xpb -qmr -Drr -s2 -b2n0 -uDD -F -Sx
Static: cl+proj-wR(∗,3)C :: -f [file] -k -Tmfrstonenosp -xpb -qmr -Drr -s3 -b2n0 -uDD -F -Sx
Static: cl+proj-wR(∗,4)C :: -f [file] -k -Tmfrstonenosp -xpb -qmr -Drr -s4 -b2n0 -uDD -F -Sx
Static: cl+proj-wR(∗,|C|)C :: -f [file] -k -Tmfrstonenosp -xpb -qmr -s0 -b2n0 -uDD -F -Sx


Static: cl+bin-wR(∗,2)C :: -f [file] -k -Tmfrstonebinsp+ -xpb -qmr -Drr -s2 -b2n0 -uDD -F -Sx
Static: cl+bin-wR(∗,3)C :: -f [file] -k -Tmfrstonebinsp+ -xpb -qmr -Drr -s3 -b2n0 -uDD -F -Sx
Static: cl+bin-wR(∗,4)C :: -f [file] -k -Tmfrstonebinsp+ -xpb -qmr -Drr -s4 -b2n0 -uDD -F -Sx
Static: cl+bin-wR(∗,|C|)C :: -f [file] -k -Tmfrstonebinsp+ -xpb -qmr -s0 -b2n0 -uDD -F -Sx


Static: cl+clq-wR(∗,2)C :: -f [file] -k -Tmfrstonemaxcsp+ -xpb -qmr -Drr -s2 -b2n0 -uDD -F -Sx
Static: cl+clq-wR(∗,2)C :: -f [file] -k -Tmfrstonemaxcsp+ -xpb -qmr -Drr -s3 -b2n0 -uDD -F -Sx
Static: cl+clq-wR(∗,2)C :: -f [file] -k -Tmfrstonemaxcsp+ -xpb -qmr -Drr -s4 -b2n0 -uDD -F -Sx
Static: cl+clq-wR(∗,|C|)C :: -f [file] -k -Tmfrstonemaxcsp+ -xpb -qmr -s0 -b2n0 -uDD -F -Sx


Priority: cl+proj-wR(∗,2)C :: -f [file] -k -Tmfrstonenosp -xss -qm12 -Drr -s2 -b2n2 -uDD -F -Sx’:
Priority: cl+proj-wR(∗,3)C :: -f [file] -k -Tmfrstonenosp -xss -qm12 -Drr -s3 -b2n2 -uDD -F -Sx’:
Priority: cl+proj-wR(∗,4)C :: -f [file] -k -Tmfrstonenosp -xss -qm12 -Drr -s4 -b2n2 -uDD -F -Sx’:
Priority: cl-wR(∗,|C|)C :: -f [file] -k -Tmfrstonenosp -xss -qm12 -s0 -b2n2 -uDD -F -Sx


Priority: cl+proj-wR(∗,2)C :: -f [file] -k -Tmfrstonenosp -xpb -qm12 -Drr -s2 -b2n2 -uDD -F -Sx
Priority: cl+proj-wR(∗,3)C :: -f [file] -k -Tmfrstonenosp -xpb -qm12 -Drr -s3 -b2n2 -uDD -F -Sx
Priority: cl+proj-wR(∗,4)C :: -f [file] -k -Tmfrstonenosp -xpb -qm12 -Drr -s4 -b2n2 -uDD -F -Sx
Priority: cl+proj-wR(∗,|C|)C :: -f [file] -k -Tmfrstonenosp -xpb -qm12 -s0 -b2n2 -uDD -F -Sx


Dynamic: cl-wR(∗,2)C :: -f [file] -f -k -Tmfrstonenosp -xss -qm122 -Drr -s2 -b2n2 -uDD -F -Sx’:
Dynamic: cl-wR(∗,3)C :: -f [file] -f -k -Tmfrstonenosp -xss -qm122 -Drr -s3 -b2n2 -uDD -F -Sx’:
Dynamic: cl-wR(∗,4)C :: -f [file] -f -k -Tmfrstonenosp -xss -qm122 -Drr -s4 -b2n2 -uDD -F -Sx’:
Dynamic: cl-wR(∗,|C|)C :: -f [file] -k -Tmfrstonenosp -xss -qm122 -s0 -b2n2 -uDD -F -Sx


Dynamic: cl+proj-wR(∗,2)C :: -f [file] -k -Tmfrstonenosp -xpb -qm122 -Drr -s2 -b2n2 -uDD -F -Sx
Dynamic: cl+proj-wR(∗,3)C :: -f [file] -k -Tmfrstonenosp -xpb -qm122 -Drr -s3 -b2n2 -uDD -F -Sx
Dynamic: cl+proj-wR(∗,4)C :: -f [file] -k -Tmfrstonenosp -xpb -qm122 -Drr -s4 -b2n2 -uDD -F -Sx
Dynamic: cl+proj-wR(∗,|C|)C :: -f [file] -f -k -Tmfrstonenosp -xpb -qm122 -s0 -b2n2 -uDD -F -Sx
```

Minimal Network of Each Cluster

PerTuple :: -f [file] -k -F -v-10 -Tmfrstonenosp -s0 -xpb -qmr -b2x0
AllSol :: -f [file] -k -F -v-10 -Tmfrstallnosp -s0 -xpb -qmr -b2x0
Features :: -f [file] -k -F -v-10 -Tmfrstnosp -s0 -xpb -qmr -b2x0

Minimal Network of the problem (considered as a single clique)

PerTuple :: if [file] -k -F -v-10 -Tmfrstonenosp -s0 -xpb -b1x0 -Sx
AllSol :: if [file] -k -F -v-10 -Tmfrstallnosp -s0 -xpb -b1x0 -Sx

Finding All solutions BTD:

GAC :: -f [file] -k -Ubtd -uDD -DGA -F -Sall -v-1

wR(∗,2)C :: -f [file] -Ubtd -k -Tmfrstonenosp -xpb -Drr -s2 -b1 -uDD -F -Sall -v-1
wR(∗,3)C :: -f [file] -Ubtd -k -Tmfrstonenosp -xpb -Drr -s3 -b1 -uDD -F -Sall -v-1
wR(∗,4)C :: -f [file] -Ubtd -k -Tmfrstonenosp -xpb -Drr -s4 -b1 -uDD -F -Sall -v-1

Static: cl-wR(∗,2)C :: -f [file] -Ubtd -k -Tmfrstonenosp -xss -qmr -Drr -s2 -b2n0 -uDD -F -Sall
Static: cl-wR(∗,3)C :: -f [file] -Ubtd -k -Tmfrstonenosp -xss -qmr -Drr -s3 -b2n0 -uDD -F -Sall
Static: cl-wR(∗,4)C :: -f [file] -Ubtd -k -Tmfrstonenosp -xss -qmr -Drr -s4 -b2n0 -uDD -F -Sall
Static: cl-wR(∗,|C|)C :: -f [file] -Ubtd -k -Tmfrstonenosp -xss -qmr -s0 -b2n0 -uDD -F -Sall

Static: cl+proj-wR(∗,2)C :: -f [file] -Ubtd -k -Tmfrstonenosp -xpb -qmr -Drr -s2 -b2n0 -uDD -F -Sall
Static: cl+proj-wR(∗,3)C :: -f [file] -Ubtd -k -Tmfrstonenosp -xpb -qmr -Drr -s3 -b2n0 -uDD -F -Sall
Static: cl+proj-wR(∗,4)C :: -f [file] -Ubtd -k -Tmfrstonenosp -xpb -qmr -Drr -s4 -b2n0 -uDD -F -Sall
Static: cl+proj-wR(∗,|C|)C :: -f [file] -Ubtd -k -Tmfrstonenosp -xpb -qmr -s0 -b2n0 -uDD -F -Sall

Static: cl+bin-wR(∗,2)C :: -f [file] -Ubtd -k -Tmfrstonebinsp+ -xpb -qmr -Drr -s2 -b2n0 -uDD -F -Sall
Static: cl+bin-wR(∗,3)C :: -f [file] -Ubtd -k -Tmfrstonebinsp+ -xpb -qmr -Drr -s3 -b2n0 -uDD -F -Sall
Static: cl+bin-wR(∗,4)C :: -f [file] -Ubtd -k -Tmfrstonebinsp+ -xpb -qmr -Drr -s4 -b2n0 -uDD -F -Sall
Static: cl+bin-wR(∗,|C|)C :: -f [file] -Ubtd -k -Tmfrstonebinsp+ -xpb -qmr -s0 -b2n0 -uDD -F -Sall

Static: cl+clq-wR(∗,2)C :: -f [file] -Ubtd -k -Tmfrstonemaxcsp+ -xpb -qmr -Drr -s2 -b2n0 -uDD -F -Sall
Static: cl+clq-wR(∗,2)C :: -f [file] -Ubtd -k -Tmfrstonemaxcsp+ -xpb -qmr -Drr -s3 -b2n0 -uDD -F -Sall
Static: cl+clq-wR(∗,2)C :: -f [file] -Ubtd -k -Tmfrstonemaxcsp+ -xpb -qmr -Drr -s4 -b2n0 -uDD -F -Sall
Static: cl+clq-wR(∗,|C|)C :: -f [file] -Ubtd -k -Tmfrstonemaxcsp+ -xpb -qmr -s0 -b2n0 -uDD -F -Sall

Finding All solutions witnessBTD:

GAC :: -f [file] -k -Uwbtd -uDD -DGA -F -Sall -v-1

wR(∗,2)C :: -f [file] -Uwbtd -k -Tmfrstonenosp -xpb -Drr -s2 -b1 -uDD -F -Sall -v-1
wR(∗,3)C :: -f [file] -Uwbtd -k -Tmfrstonenosp -xpb -Drr -s3 -b1 -uDD -F -Sall -v-1
wR(∗,4)C :: -f [file] -Uwbtd -k -Tmfrstonenosp -xpb -Drr -s4 -b1 -uDD -F -Sall -v-1

Static: cl-wR(∗,2)C :: -f [file] -Uwbtd -k -Tmfrstonenosp -xss -qmr -Drr -s2 -b2n0 -uDD -F -Sall
Static: cl-wR(∗,3)C :: -f [file] -Uwbtd -k -Tmfrstonenosp -xss -qmr -Drr -s3 -b2n0 -uDD -F -Sall
Static: cl-wR(∗,4)C :: -f [file] -Uwbtd -k -Tmfrstonenosp -xss -qmr -Drr -s4 -b2n0 -uDD -F -Sall
Static: cl-wR(∗,|C|)C :: -f [file] -Uwbtd -k -Tmfrstonenosp -xss -qmr -s0 -b2n0 -uDD -F -Sall

Static: cl+proj-wR(∗,2)C :: -f [file] -Uwbtd -k -Tmfrstonenosp -xpb -qmr -Drr -s2 -b2n0 -uDD -F -Sall
Static: cl+proj-wR(∗,3)C :: -f [file] -Uwbtd -k -Tmfrstonenosp -xpb -qmr -Drr -s3 -b2n0 -uDD -F -Sall
Static: cl+proj-wR(∗,4)C :: -f [file] -Uwbtd -k -Tmfrstonenosp -xpb -qmr -Drr -s4 -b2n0 -uDD -F -Sall
Static: cl+proj-wR(∗,|C|)C :: -f [file] -Uwbtd -k -Tmfrstonenosp -xpb -qmr -s0 -b2n0 -uDD -F -Sall

Static: cl+bin-wR(∗,2)C :: -f [file] -Uwbtd -k -Tmfrstonebinsp+ -xpb -qmr -Drr -s2 -b2n0 -uDD -F -Sall
Static: cl+bin-wR(∗,3)C :: -f [file] -Uwbtd -k -Tmfrstonebinsp+ -xpb -qmr -Drr -s3 -b2n0 -uDD -F -Sall
Static: cl+bin-wR(∗,4)C :: -f [file] -Uwbtd -k -Tmfrstonebinsp+ -xpb -qmr -Drr -s4 -b2n0 -uDD -F -Sall
Static: cl+bin-wR(∗,|C|)C :: -f [file] -Uwbtd -k -Tmfrstonebinsp+ -xpb -qmr -s0 -b2n0 -uDD -F -Sall

Static: cl+clq-wR(∗,2)C :: -f [file] -Uwbtd -k -Tmfrstonemaxcsp+ -xpb -qmr -Drr -s2 -b2n0 -uDD -F -Sall
Static: cl+clq-wR(∗,2)C :: -f [file] -Uwbtd -k -Tmfrstonemaxcsp+ -xpb -qmr -Drr -s3 -b2n0 -uDD -F -Sall
Static: cl+clq-wR(∗,2)C :: -f [file] -Uwbtd -k -Tmfrstonemaxcsp+ -xpb -qmr -Drr -s4 -b2n0 -uDD -F -Sall
Static: cl+clq-wR(∗,|C|)C :: -f [file] -Uwbtd -k -Tmfrstonemaxcsp+ -xpb -qmr -s0 -b2n0 -uDD -F -Sall

4.37.2 Function Documentation

4.37.2.1 void check_ok (**main_structure** ∗ *m_s*)

4.37.2.2 void check_ok2 (**variables** ∗ *vars*)

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

4.37.2.4 void sigxcpu_handler (int *signo*)

4.37.3 Variable Documentation

4.37.3.1 **main_structure**∗ m_s_g

4.38 minibucket.c File Reference

```
#include "minibucket.h"
#include "rs2.h"
#include "rs.h"
#include "decomposition_functions.h"
#include "hashtable_search.h"
```

Functions

- int partition_bucket (decomposed_tree ∗treen)
- htable ∗ mbe_join_tables (htable ∗hta, htable ∗htb)

4.38.1 Function Documentation

4.38.1.1 htable∗ mbe_join_tables (htable ∗ *hta,* htable ∗ *htb*)

4.38.1.2 int partition_bucket (decomposed_tree ∗ *treen*)

4.39 minibucket.h File Reference

```
#include "utils.h"
#include "decomposition.h"
#include "llist.h"
```

Functions

- int partition_bucket (decomposed_tree ∗treen)
- int partition_bucket_old (decomposed_tree ∗treen)
- htable ∗ mbe_join_tables (htable ∗hta, htable ∗htb)

4.39.1 Detailed Description

Implements the mini-bucket algorithm.

4.39.2 Function Documentation

4.39.2.1 htable∗ mbe_join_tables (htable ∗ *hta,* htable ∗ *htb*)

4.39.2.2 int partition_bucket (decomposed_tree ∗ *treen*)

4.39.2.3 int partition_bucket_old (decomposed_tree ∗ *treen*)

4.40 oneall.c File Reference

```
#include "oneall.h"
#include "utils.h"
```

Functions

- int evaluate_on_tree (decision_tree ∗dt, rs2tree_stats ∗rss)
- decision_tree ∗ get_dt_node (char ∗attribute, double value, char ∗∗names)
- decision_tree_system ∗ parse_tree (void)
- void print_decision_tree (decision_tree ∗dt)

4.40.1 Function Documentation

4.40.1.1 int evaluate_on_tree (decision_tree ∗ *dt,* rs2tree_stats ∗ *rss*)

4.40.1.2 decision_tree∗ get_dt_node (char ∗ *attribute,* double *value,* char ∗∗ *names*)

4.40.1.3 decision_tree_system∗ parse_tree (void)

4.40.1.4 void print_decision_tree (decision_tree ∗ *dt*)

4.41 oneall.h File Reference

```
#include "rs2tree.h"
```

Classes

- struct decision_tree
- struct decision_tree_system

Typedefs

- typedef struct decision_tree decision_tree
- typedef struct decision_tree_system decision_tree_system

Functions

- int evaluate_on_tree (decision_tree ∗dt, rs2tree_stats ∗rss)
- decision_tree ∗ get_dt_node (char ∗attribute, double value, char ∗∗names)
- decision_tree_system ∗ parse_tree (void)
- void print_decision_tree (decision_tree ∗dt)

4.41.1 Detailed Description

Provides the algorithm for parsing a decision tree for choosing between PerTuple and AllSol algorithms.

4.41.2 Typedef Documentation

4.41.2.1 typedef struct **decision_tree decision_tree**

4.41.2.2 typedef struct **decision_tree_system decision_tree_system**

4.41.3 Function Documentation

4.41.3.1 int evaluate_on_tree (**decision_tree** ∗ *dt,* **rs2tree_stats** ∗ *rss*)

4.41.3.2 **decision_tree**∗ get_dt_node (char ∗ *attribute,* double *value,* char ∗∗ *names*)

4.41.3.3 **decision_tree_system**∗ parse_tree (void)

4.41.3.4 void print_decision_tree (**decision_tree** ∗ *dt*)

4.42 parsers.c File Reference

```
#include "parsers.h"
#include "query_graph.h"
#include "constraints.h"
#include "decomposition.h"
#include "decomposition_functions.h"
#include <math.h>
```

Functions

- void read_uai (FILE ∗file, main_structure ∗main_struct, double weight_th)
- void read_ergos (FILE ∗file, main_structure ∗main_struct, double weight_th)
- void unrank (int ∗vars, int ∗tuple, int rank)
- void read_comment (FILE ∗file)
- void read_wcsp (FILE ∗file, main_structure ∗main_struct)

4.42.1 Function Documentation

4.42.1.1 void read_comment (FILE ∗ *file*)

4.42.1.2 void read_ergos (FILE ∗ *file,* main_structure ∗ *main_struct,* double *weight_th*)

4.42.1.3 void read_uai (FILE ∗ *file,* main_structure ∗ *main_struct,* double *weight_th*)

4.42.1.4 void read_wcsp (FILE ∗ *file,* main_structure ∗ *main_struct*)

4.42.1.5 void unrank (int ∗ *vars,* int ∗ *tuple,* int *rank*)

4.43 parsers.h File Reference

```
#include "utils.h"
#include "stdio.h"
```

Functions

- void read_uai (FILE ∗file, main_structure ∗main_struct, double weight_th)
- void read_ergos (FILE ∗file, main_structure ∗main_struct, double weight_th)
- void read_wcsp (FILE ∗file, main_structure ∗main_struct)
- void read_comment (FILE ∗file)
- void unrank (int ∗vars, int ∗tuple, int rank)

4.43.1 Detailed Description

Implements parsers for various CSP representations.

4.43.2 Function Documentation

4.43.2.1 void read_comment (FILE ∗ *file*)

4.43.2.2 void read_ergos (FILE ∗ *file,* main_structure ∗ *main_struct,* double *weight_th*)

4.43.2.3 void read_uai (FILE ∗ *file,* main_structure ∗ *main_struct,* double *weight_th*)

4.43.2.4 void read_wcsp (FILE ∗ *file,* main_structure ∗ *main_struct*)

4.43.2.5 void unrank (int ∗ *vars,* int ∗ *tuple,* int *rank*)

4.44 partition.c File Reference

```
#include "partition.h"
#include "utils.h"
#include "decomposition.h"
#include "llist.h"
#include "hashtable_search.h"
```

Functions

- llist * partition_separator (decomposed_tree *treen, int *varmap)
- llist * binary_rel_separator (decomposed_tree *treen, int *varmap)
- llist * all_binary_rel_separator (decomposed_tree *treen, int *varmap)

4.44.1 Function Documentation

4.44.1.1 llist* all_binary_rel_separator (**decomposed_tree** * *treen,* int * *varmap*)

4.44.1.2 llist* binary_rel_separator (**decomposed_tree** * *treen,* int * *varmap*)

4.44.1.3 llist* partition_separator (**decomposed_tree** * *treen,* int * *varmap*)

4.45 partition.h File Reference

```
#include "llist.h"
#include "decomposition.h"
```

Functions

- llist * partition_separator (decomposed_tree *treen, int *varmap)
- llist * binary_rel_separator (decomposed_tree *treen, int *varmap)
- llist * all_binary_rel_separator (decomposed_tree *treen, int *varmap)

4.45.1 Detailed Description

Implements helper methods for generating redundant constraints at the separators.

4.45.2 Function Documentation

4.45.2.1 llist* all_binary_rel_separator (**decomposed_tree** * *treen,* int * *varmap*)

4.45.2.2 llist* binary_rel_separator (**decomposed_tree** * *treen,* int * *varmap*)

4.45.2.3 llist* partition_separator (**decomposed_tree** * *treen,* int * *varmap*)

4.46 query_graph.c File Reference

```
#include <stdio.h>
#include "query_graph.h"
#include "llist.h"
#include "utils.h"
#include "tree_decomp.h"
#include "rs.h"
#include "rs2.h"
#include "rpwc.h"
#include "tree.h"
#include "constraints.h"
#include "relations.h"
#include "rpwcprocedures.h"
#include "rsprocedures.h"
#include "sort.h"
#include "combinations.h"
#include "db_algo1.h"
#include "rr.h"
```

Functions

- void delete_node ()
- constraint_graph ∗ new_constraint_graph (int cons_count)
- void destroy_constraint_graph (constraint_graph ∗cg)
- constraint_graph_node ∗ new_constraint_graph_node_t (htable ∗ht)
- constraint_graph_node ∗ new_constraint_graph_node (constraint ∗cons)
- void destroy_constraint_graph_node (constraint_graph_node ∗cgn)
- constraint_graph_edge ∗ new_constraint_graph_edge (constraint_graph_node ∗node, constraint_graph_node ∗neighbour, constraint_graph ∗cg)
- void destroy_constraint_graph_edge (constraint_graph_edge ∗cge)
- constraint_graph_node ∗ get_other_end (constraint_graph_node ∗node, constraint_graph_edge ∗edge)
- constraint_graph_edge ∗ get_edge (constraint_graph_node ∗node1, constraint_graph_node ∗node2, constraint_graph ∗cg)
- constraint_graph_node ∗ get_constraint_graph_node (constraint ∗cons, constraint_graph ∗c_graph)
- constraint_graph_node ∗ get_node_with_constraint_id (int id, constraint_graph ∗cg)
- constraint_graph_node ∗ add_constraint_to_graph (constraint ∗cons, constraint_graph ∗c_graph)
- constraint_graph_node ∗ add_constraint_to_graph_force (htable ∗ht, constraint_graph ∗c_graph)
- constraint_graph_node ∗∗ expand_cgn_i (constraint_graph ∗c_graph)
- constraint_graph_node ∗ add_or_get_constraint_node (constraint ∗cons, constraint_graph ∗c_graph)
- int get_value_edge_sort (void ∗e, int row, int column)
- void ∗ get_object_edge_sort (void ∗e, int row, int column)
- void set_object_edge_sort (void ∗e, int row, int column, void ∗value)
- char ∗∗ build_adjacency_matrix (constraint_graph ∗cg)
- void build_adjacency_matrix2 (constraint_graph ∗cg)
- constraint_graph ∗ init_graph (main_structure ∗m_s)
- constraint_graph ∗ init_graph_with_fill_edges (main_structure ∗m_s, constraints ∗cons, llist ∗fill_edges, int silent)
- int compare_scope_cge_global_list (const void ∗o1, const void ∗o2)
- int compare_scope_cge (const void ∗o1, const void ∗o2)
- constraint_graph ∗ init_graph_given_constraint_set (main_structure ∗m_s, set ∗constraints, int silent)
- void add_neighbours_for_node (constraint_graph_node ∗c_node, constraint ∗seled_const, constraint_graph ∗c_graph, main_structure ∗m_s)

- void add_neighbours_for_node_given_constraint_set (constraint_graph_node ∗c_node, constraint ∗seled_const, constraint_graph ∗c_graph, set ∗constraints, main_structure ∗m_s)
- void add_neighbours_for_fill_edge (int ∗fill_edge, constraint_graph ∗c_graph, main_structure ∗m_s)
- void add_neighbours_for_fill_edge_given_constraints_set (int ∗fill_edge, constraint_graph ∗c_graph, set ∗constraints, main_structure ∗m_s)
- void check_connected (constraint_graph_node ∗initial_node, int ∗constraints, int ∗variables)
- void set_weights (constraint_graph ∗c_g)
- void print_number_of_deleted_tuples (constraint_graph ∗g)
- void print_graph (constraint_graph ∗g, int print_relations)
- void print_graph_scopes (constraint_graph ∗g, int print_relations)
- void print_tree (constraint_graph_node ∗root, int indent)
- void reset_colors (constraint_graph ∗g)
- constraint_graph_node ∗ get_initial_node (constraint_graph ∗g)
- int measure (constraint_graph_node ∗to)
- void update_constraint_tuples (constraints ∗cons)
- void evaluate_edges (constraint_graph ∗cg, int limit)
- int compare (void ∗value, void ∗bodyb)
- int get_cnode_id (void ∗obj)
- int merge_nodes (constraint_graph_node ∗nodea, constraint_graph_node ∗nodeb, constraint_graph ∗cg, int limit)
- void remove_node_from_graph (constraint_graph_node ∗node, constraint_graph ∗cg)
- void remove_edge_from_graph (constraint_graph_edge ∗edge, constraint_graph ∗cg)
- q_node ∗ build_plan (constraint_graph_node ∗root)
- int get_estimated_value_weight (void ∗body)
- void merge_nodes_no_join (constraint_graph_node ∗nodea, constraint_graph_node ∗nodeb, constraint_graph ∗cg)
- int has_common_cols (int ∗var1, int ∗var2, int ∗vmap)
- void append_to_nodes_combination (constraint_graph ∗cg, constraint_graph ∗notin_cg, llist ∗toadd)
- int get_all_nodes_combination (constraint_graph ∗cg, llist ∗combs)
- void printGraphNeato (constraint_graph ∗cg, const char ∗fileName)
- void printGraphNeato2 (constraint_graph ∗cg)
- constraint_graph ∗ add_constraints_to_graph (llist ∗list_of_binary_constraint_vars, constraint_graph ∗cg)
- void remove_fill_edges_from_variables (variables ∗vars)
- rbt_set ∗ new_neighbour_set (int constraint_count, int minimal)

Variables

- main_structure ∗ m_s_g

4.46.1 Function Documentation

4.46.1.1 constraint_graph_node∗ add_constraint_to_graph (constraint ∗ *cons,* constraint_graph ∗ *c_graph*)

4.46.1.2 constraint_graph_node∗ add_constraint_to_graph_force (htable ∗ *ht,* constraint_graph ∗ *c_graph*)

4.46.1.3 constraint_graph∗ add_constraints_to_graph (llist ∗ *list_of_binary_constraint_vars,* constraint_graph ∗ *cg*)

4.46.1.4 void add_neighbours_for_fill_edge (int ∗ *fill_edge,* constraint_graph ∗ *c_graph,* main_structure ∗ *m_s*)

4.46.1.5 void add_neighbours_for_fill_edge_given_constraints_set (int ∗ *fill_edge,* constraint_graph ∗ *c_graph,* set ∗ *constraints,* main_structure ∗ *m_s*)

4.46.1.6 void add_neighbours_for_node (constraint_graph_node ∗ *c_node,* constraint ∗ *seled_const,* constraint_graph ∗ *c_graph,* main_structure ∗ *m_s*)

4.46.1.7 void add_neighbours_for_node_given_constraint_set (constraint_graph_node * c_node, constraint * seled_const, constraint_graph * c_graph, set * constraints, main_structure * m_s)

4.46.1.8 constraint_graph_node* add_or_get_constraint_node (constraint * cons, constraint_graph * c_graph)

4.46.1.9 void append_to_nodes_combination (constraint_graph * cg, constraint_graph * notin_cg, llist * toadd)

4.46.1.10 char** build_adjacency_matrix (constraint_graph * cg)

4.46.1.11 void build_adjacency_matrix2 (constraint_graph * cg)

4.46.1.12 q_node* build_plan (constraint_graph_node * root)

4.46.1.13 void check_connected (constraint_graph_node * initial_node, int * constraints, int * variables)

4.46.1.14 int compare (void * value, void * bodyb)

4.46.1.15 int compare_scope_cge (const void * o1, const void * o2)

4.46.1.16 int compare_scope_cge_global_list (const void * o1, const void * o2)

4.46.1.17 void delete_node ( )

4.46.1.18 void destroy_constraint_graph (constraint_graph * cg)

4.46.1.19 void destroy_constraint_graph_edge (constraint_graph_edge * cge)

4.46.1.20 void destroy_constraint_graph_node (constraint_graph_node * cgn)

4.46.1.21 void evaluate_edges (constraint_graph * cg, int limit)

4.46.1.22 constraint_graph_node** expand_cgn_i (constraint_graph * c_graph)

4.46.1.23 int get_all_nodes_combination (constraint_graph * cg, llist * combs)

4.46.1.24 int get_cnode_id (void * obj)

4.46.1.25 constraint_graph_node* get_constraint_graph_node (constraint * cons, constraint_graph * c_graph)

4.46.1.26 constraint_graph_edge* get_edge (constraint_graph_node * node1, constraint_graph_node * node2, constraint_graph * cg)

4.46.1.27 int get_estimated_value_weight (void * body)

4.46.1.28 constraint_graph_node* get_initial_node (constraint_graph * g)

4.46.1.29 constraint_graph_node* get_node_with_constraint_id (int id, constraint_graph * cg)

4.46.1.30 void* get_object_edge_sort (void * e, int row, int column)

4.46.1.31 constraint_graph_node* get_other_end (constraint_graph_node * node, constraint_graph_edge * edge)

4.46.1.32 int get_value_edge_sort (void * e, int row, int column)

4.46.1.33 int has_common_cols (int * var1, int * var2, int * vmap)

4.46.1.34 constraint_graph∗ init_graph (main_structure ∗ *m_s*)

4.46.1.35 constraint_graph∗ init_graph_given_constraint_set (main_structure ∗ *m_s,* set ∗ *constraints,* int *silent*)

4.46.1.36 constraint_graph∗ init_graph_with_fill_edges (main_structure ∗ *m_s,* constraints ∗ *cons,* llist ∗ *fill_edges,* int *silent*)

4.46.1.37 int measure (constraint_graph_node ∗ *to*)

4.46.1.38 int merge_nodes (constraint_graph_node ∗ *nodea,* constraint_graph_node ∗ *nodeb,* constraint_graph ∗ *cg,* int *limit*)

4.46.1.39 void merge_nodes_no_join (constraint_graph_node ∗ *nodea,* constraint_graph_node ∗ *nodeb,* constraint_graph ∗ *cg*)

4.46.1.40 constraint_graph∗ new_constraint_graph (int *cons_count*)

4.46.1.41 constraint_graph_edge∗ new_constraint_graph_edge (constraint_graph_node ∗ *node,* constraint_graph_node ∗ *neighbour,* constraint_graph ∗ *cg*)

4.46.1.42 constraint_graph_node∗ new_constraint_graph_node (constraint ∗ *cons*)

4.46.1.43 constraint_graph_node∗ new_constraint_graph_node_t (htable ∗ *ht*)

4.46.1.44 rbt_set∗ new_neighbour_set (int *constraint_count,* int *minimal*)

4.46.1.45 void print_graph (constraint_graph ∗ *g,* int *print_relations*)

4.46.1.46 void print_graph_scopes (constraint_graph ∗ *g,* int *print_relations*)

4.46.1.47 void print_number_of_deleted_tuples (constraint_graph ∗ *g*)

4.46.1.48 void print_tree (constraint_graph_node ∗ *root,* int *indent*)

4.46.1.49 void printGraphNeato (constraint_graph ∗ *cg,* const char ∗ *fileName*)

4.46.1.50 void printGraphNeato2 (constraint_graph ∗ *cg*)

4.46.1.51 void remove_edge_from_graph (constraint_graph_edge ∗ *edge,* constraint_graph ∗ *cg*)

4.46.1.52 void remove_fill_edges_from_variables (variables ∗ *vars*)

4.46.1.53 void remove_node_from_graph (constraint_graph_node ∗ *node,* constraint_graph ∗ *cg*)

4.46.1.54 void reset_colors (constraint_graph ∗ *g*)

4.46.1.55 void set_object_edge_sort (void ∗ *e,* int *row,* int *column,* void ∗ *value*)

4.46.1.56 void set_weights (constraint_graph ∗ *c_g*)

4.46.1.57 void update_constraint_tuples (constraints ∗ *cons*)

4.46.2 Variable Documentation

4.46.2.1 main_structure∗ m_s_g

4.47 query_graph.h File Reference

```
#include "globals.h"
#include "constraints.h"
#include "variables.h"
#include "hashtable.h"
#include "query_plan.h"
#include "llist.h"
#include "rbt_set.h"
```

Classes

- struct constraint_graph_node
- struct constraint_graph_edge
- struct constraint_graph

Macros

- #define INITIAL_LIMIT 100

Typedefs

- typedef struct constraint_graph_node constraint_graph_node
- typedef struct constraint_graph_edge constraint_graph_edge
- typedef struct constraint_graph constraint_graph

Functions

- char ∗∗ build_adjacency_matrix (constraint_graph ∗cg)
- void build_adjacency_matrix2 (constraint_graph ∗cg)
- void destroy_constraint_graph (constraint_graph ∗cg)
- void destroy_constraint_graph_node (constraint_graph_node ∗cgn)
- void destroy_constraint_graph_edge (constraint_graph_edge ∗cge)
- constraint_graph_node ∗∗ expand_cgn_i (constraint_graph ∗c_graph)
- void check_connected (constraint_graph_node ∗initial_node, int ∗constrinats, int ∗variables)
- constraint_graph ∗ new_constraint_graph (int cons_count)
- constraint_graph_node ∗ new_constraint_graph_node (constraint ∗cons)
- constraint_graph_edge ∗ new_constraint_graph_edge (constraint_graph_node ∗node, constraint_graph_node ∗neighbour, constraint_graph ∗cg)
- constraint_graph_node ∗ get_other_end (constraint_graph_node ∗node, constraint_graph_edge ∗edge)
- constraint_graph_edge ∗ get_edge (constraint_graph_node ∗node1, constraint_graph_node ∗node2, constraint_graph ∗cg)
- constraint_graph_node ∗ get_constraint_graph_node (constraint ∗cons, constraint_graph ∗c_graph)
- constraint_graph_node ∗ add_constraint_to_graph (constraint ∗cons, constraint_graph ∗c_graph)
- constraint_graph_node ∗ add_constraint_to_graph_force (htable ∗ht, constraint_graph ∗c_graph)
- constraint_graph_node ∗ add_or_get_constraint_node (constraint ∗cons, constraint_graph ∗c_graph)
- constraint_graph ∗ init_graph (main_structure ∗m_s)
- void set_weights (constraint_graph ∗c_g)
- void print_number_of_deleted_tuples (constraint_graph ∗g)
- void print_graph (constraint_graph ∗g, int print_relations)

- void print_graph_scopes (constraint_graph ∗g, int print_relations)
- void print_tree (constraint_graph_node ∗root, int indent)
- void reset_colors (constraint_graph ∗g)
- constraint_graph_node ∗ get_initial_node (constraint_graph ∗g)
- int measure (constraint_graph_node ∗to)
- void make_tree (constraint_graph_node ∗initial_node, int measure(void))
- void greedy_naive (constraint_graph ∗c_graph)
- int do_db (main_structure ∗m_s)
- void evaluate_edges (constraint_graph ∗cg, int limit)
- void update_constraint_tuples (constraints ∗cons)
- void remove_node_from_graph (constraint_graph_node ∗node, constraint_graph ∗cg)
- void remove_edge_from_graph (constraint_graph_edge ∗edge, constraint_graph ∗cg)
- int get_value_edge_sort (void ∗e, int row, int column)
- void ∗ get_object_edge_sort (void ∗e, int row, int column)
- void set_object_edge_sort (void ∗e, int row, int column, void ∗value)
- constraint_graph_node ∗ get_node_with_constraint_id (int id, constraint_graph ∗cg)
- int compare (void ∗value, void ∗bodyb)
- int merge_nodes (constraint_graph_node ∗nodea, constraint_graph_node ∗nodeb, constraint_graph ∗cg, int limit)
- q_node ∗ build_plan (constraint_graph_node ∗root)
- int get_estimated_value_weight (void ∗body)
- q_node ∗ buld_join_tree (constraint_graph ∗c_graph)
- void merge_nodes_no_join (constraint_graph_node ∗nodea, constraint_graph_node ∗nodeb, constraint_graph ∗cg)
- void delete_node ()
- llist ∗ get_int_common_cols (int ∗var1, int ∗var2, int ∗vmap)
- int has_common_cols (int ∗var1, int ∗var2, int ∗vmap)
- int get_all_nodes_combination (constraint_graph ∗cg, llist ∗combs)
- void append_to_nodes_combination (constraint_graph ∗cg, constraint_graph ∗notin_cg, llist ∗toadd)
- void printGraphNeato (constraint_graph ∗cg, const char ∗fileName)
- void printGraphNeato2 (constraint_graph ∗cg)
- constraint_graph ∗ init_graph_given_constraint_set (main_structure ∗m_s, set ∗constraints, int silent)
- constraint_graph ∗ init_graph_with_fill_edges (main_structure ∗m_s, constraints ∗cons, llist ∗fill_edges, int silent)
- void add_neighbours_for_node (constraint_graph_node ∗c_node, constraint ∗seled_const, constraint_graph ∗c_graph, main_structure ∗m_s)
- void add_neighbours_for_node_given_constraint_set (constraint_graph_node ∗c_node, constraint ∗seled_const, constraint_graph ∗c_graph, set ∗constraints, main_structure ∗m_s)
- void add_neighbours_for_fill_edge (int ∗fill_edge, constraint_graph ∗c_graph, main_structure ∗m_s)
- void add_neighbours_for_fill_edge_given_constraints_set (int ∗fill_edge, constraint_graph ∗c_graph, set ∗constraints, main_structure ∗m_s)
- constraint_graph ∗ add_constraints_to_graph (llist ∗list_of_binary_constraint_vars, constraint_graph ∗cg)
- void remove_fill_edges_from_variables (variables ∗vars)
- rbt_set ∗ new_neighbour_set (int constraint_count, int minimal)
- int get_cnode_id (void ∗obj)
- constraint_graph_node ∗ new_constraint_graph_node_t (htable ∗ht)
- int compare_scope_cge (const void ∗o1, const void ∗o2)
- int compare_scope_cge_global_list (const void ∗o1, const void ∗o2)

4.47.1 Detailed Description

Implements the dual graph and methods for manipulating them.

4.47.2 Macro Definition Documentation

4.47.2.1 #define INITIAL_LIMIT 100

4.47.3 Typedef Documentation

4.47.3.1 typedef struct constraint_graph constraint_graph

4.47.3.2 typedef struct constraint_graph_edge constraint_graph_edge

4.47.3.3 typedef struct constraint_graph_node constraint_graph_node

4.47.4 Function Documentation

4.47.4.1 constraint_graph_node∗ add_constraint_to_graph (constraint ∗ *cons,* constraint_graph ∗ *c_graph*)

4.47.4.2 constraint_graph_node∗ add_constraint_to_graph_force (htable ∗ *ht,* constraint_graph ∗ *c_graph*)

4.47.4.3 constraint_graph∗ add_constraints_to_graph (llist ∗ *list_of_binary_constraint_vars,* constraint_graph ∗ *cg*)

4.47.4.4 void add_neighbours_for_fill_edge (int ∗ *fill_edge,* constraint_graph ∗ *c_graph,* main_structure ∗ *m_s*)

4.47.4.5 void add_neighbours_for_fill_edge_given_constraints_set (int ∗ *fill_edge,* constraint_graph ∗ *c_graph,* set ∗ *constraints,* main_structure ∗ *m_s*)

4.47.4.6 void add_neighbours_for_node (constraint_graph_node ∗ *c_node,* constraint ∗ *seled_const,* constraint_graph ∗ *c_graph,* main_structure ∗ *m_s*)

4.47.4.7 void add_neighbours_for_node_given_constraint_set (constraint_graph_node ∗ *c_node,* constraint ∗ *seled_const,* constraint_graph ∗ *c_graph,* set ∗ *constraints,* main_structure ∗ *m_s*)

4.47.4.8 constraint_graph_node∗ add_or_get_constraint_node (constraint ∗ *cons,* constraint_graph ∗ *c_graph*)

4.47.4.9 void append_to_nodes_combination (constraint_graph ∗ *cg,* constraint_graph ∗ *notin_cg,* llist ∗ *toadd*)

4.47.4.10 char∗∗ build_adjacency_matrix (constraint_graph ∗ *cg*)

4.47.4.11 void build_adjacency_matrix2 (constraint_graph ∗ *cg*)

4.47.4.12 q_node∗ build_plan (constraint_graph_node ∗ *root*)

4.47.4.13 q_node∗ buld_join_tree (constraint_graph ∗ *c_graph*)

4.47.4.14 void check_connected (constraint_graph_node ∗ *initial_node,* int ∗ *constrinats,* int ∗ *variables*)

4.47.4.15 int compare (void ∗ *value,* void ∗ *bodyb*)

4.47.4.16 int compare_scope_cge (const void ∗ *o1,* const void ∗ *o2*)

4.47.4.17 int compare_scope_cge_global_list (const void ∗ *o1,* const void ∗ *o2*)

4.47.4.18 void delete_node ( )

4.47.4.19 void destroy_constraint_graph (constraint_graph ∗ *cg*)

4.47.4.20 void destroy_constraint_graph_edge (constraint_graph_edge ∗ *cge*)

4.47.4.21 void destroy_constraint_graph_node (constraint_graph_node * cgn)

4.47.4.22 int do_db (main_structure * m_s)

4.47.4.23 void evaluate_edges (constraint_graph * cg, int limit)

4.47.4.24 constraint_graph_node** expand_cgn_i (constraint_graph * c_graph)

4.47.4.25 int get_all_nodes_combination (constraint_graph * cg, llist * combs)

4.47.4.26 int get_cnode_id (void * obj)

4.47.4.27 constraint_graph_node* get_constraint_graph_node (constraint * cons, constraint_graph * c_graph)

4.47.4.28 constraint_graph_edge* get_edge (constraint_graph_node * node1, constraint_graph_node * node2, constraint_graph * cg)

4.47.4.29 int get_estimated_value_weight (void * body)

4.47.4.30 constraint_graph_node* get_initial_node (constraint_graph * g)

4.47.4.31 llist* get_int_common_cols (int * var1, int * var2, int * vmap)

4.47.4.32 constraint_graph_node* get_node_with_constraint_id (int id, constraint_graph * cg)

4.47.4.33 void* get_object_edge_sort (void * e, int row, int column)

4.47.4.34 constraint_graph_node* get_other_end (constraint_graph_node * node, constraint_graph_edge * edge)

4.47.4.35 int get_value_edge_sort (void * e, int row, int column)

4.47.4.36 void greedy_naive (constraint_graph * c_graph)

4.47.4.37 int has_common_cols (int * var1, int * var2, int * vmap)

4.47.4.38 constraint_graph* init_graph (main_structure * m_s)

4.47.4.39 constraint_graph* init_graph_given_constraint_set (main_structure * m_s, set * constraints, int silent)

4.47.4.40 constraint_graph* init_graph_with_fill_edges (main_structure * m_s, constraints * cons, llist * fill_edges, int silent)

4.47.4.41 void make_tree (constraint_graph_node * initial_node, int measurevoid)

4.47.4.42 int measure (constraint_graph_node * to)

4.47.4.43 int merge_nodes (constraint_graph_node * nodea, constraint_graph_node * nodeb, constraint_graph * cg, int limit)

4.47.4.44 void merge_nodes_no_join (constraint_graph_node * nodea, constraint_graph_node * nodeb, constraint_graph * cg)

4.47.4.45 constraint_graph* new_constraint_graph (int cons_count)

4.47.4.46 constraint_graph_edge∗ new_constraint_graph_edge (constraint_graph_node ∗ *node*, constraint_graph_node ∗ *neighbour*, constraint_graph ∗ *cg*)

4.47.4.47 constraint_graph_node∗ new_constraint_graph_node (constraint ∗ *cons*)

4.47.4.48 constraint_graph_node∗ new_constraint_graph_node_t (htable ∗ *ht*)

4.47.4.49 rbt_set∗ new_neighbour_set (int *constraint_count*, int *minimal*)

4.47.4.50 void print_graph (constraint_graph ∗ *g*, int *print_relations*)

4.47.4.51 void print_graph_scopes (constraint_graph ∗ *g*, int *print_relations*)

4.47.4.52 void print_number_of_deleted_tuples (constraint_graph ∗ *g*)

4.47.4.53 void print_tree (constraint_graph_node ∗ *root*, int *indent*)

4.47.4.54 void printGraphNeato (constraint_graph ∗ *cg*, const char ∗ *fileName*)

4.47.4.55 void printGraphNeato2 (constraint_graph ∗ *cg*)

4.47.4.56 void remove_edge_from_graph (constraint_graph_edge ∗ *edge*, constraint_graph ∗ *cg*)

4.47.4.57 void remove_fill_edges_from_variables (variables ∗ *vars*)

4.47.4.58 void remove_node_from_graph (constraint_graph_node ∗ *node*, constraint_graph ∗ *cg*)

4.47.4.59 void reset_colors (constraint_graph ∗ *g*)

4.47.4.60 void set_object_edge_sort (void ∗ *e*, int *row*, int *column*, void ∗ *value*)

4.47.4.61 void set_weights (constraint_graph ∗ *c_g*)

4.47.4.62 void update_constraint_tuples (constraints ∗ *cons*)

4.48 rb_tree.c File Reference

```
#include "rb_tree.h"
```

Functions

- rb_tree ∗ new_rb_tree (void ∗key, int ikey, void ∗payload)
- void destroy_rb_tree (rb_tree ∗∗root)
- void destroy_rb_tree_key (rb_tree ∗∗root, void destroyer(llist ∗))
- void destroy_rb_tree_paload (rb_tree ∗∗root, void destroyer(llist ∗))
- void rb_left_rotate (rb_tree ∗∗root, rb_tree ∗x)
- void rb_right_rotate (rb_tree ∗∗root, rb_tree ∗x)
- void ∗ rb_insert (rb_tree ∗∗root, void ∗key, int ikey, void ∗payload, int compare(int ∗, int ∗))
- void rb_insert_fixup (rb_tree ∗∗root, rb_tree ∗z)
- void ∗ rb_find (rb_tree ∗root, void ∗key, int ikey, int compare(int ∗, int ∗))
- void rb_get_sorted_order (rb_tree ∗root, int compare(int ∗, int ∗), llist ∗sorted)
- void rb_get_sorted_order_list (rb_tree ∗root, int compare(int ∗, int ∗), llist ∗sorted, llist ∗keys)
- rb_tree ∗ rb_tree_union (rb_tree ∗a, rb_tree ∗b, int compare(int ∗, int ∗))
- rb_tree ∗ rb_copy_tree (rb_tree ∗tree, int compare(int ∗, int ∗))

- void ∗ rb_delete_key (rb_tree ∗∗root, void ∗key, int ikey, int compare(int ∗, int ∗))
- void ∗ rb_delete (rb_tree ∗∗root, rb_tree ∗z)
- void rb_delete_fixup (rb_tree ∗∗root, rb_tree ∗x)
- rb_tree ∗ rb_successor (rb_tree ∗x)
- rb_tree ∗ rb_minimum (rb_tree ∗x)
- rb_tree ∗ rb_maximum (rb_tree ∗x)
- void check_tree (rb_tree ∗root)
- void rb_print_tree (rb_tree ∗root)
- void rb_print_tree_l (rb_tree ∗root, int l)

4.48.1 Function Documentation

4.48.1.1 void check_tree (**rb_tree** ∗ *root*)

4.48.1.2 void destroy_rb_tree (**rb_tree** ∗∗ *root*)

4.48.1.3 void destroy_rb_tree_key (**rb_tree** ∗∗ *root*, void *destroyerllist* ∗)

4.48.1.4 void destroy_rb_tree_paload (**rb_tree** ∗∗ *root*, void *destroyerllist* ∗)

4.48.1.5 **rb_tree**∗ new_rb_tree (void ∗ *key*, int *ikey*, void ∗ *payload*)

4.48.1.6 **rb_tree**∗ rb_copy_tree (**rb_tree** ∗ *tree*, int *compareint* ∗*, int* ∗)

4.48.1.7 void∗ rb_delete (**rb_tree** ∗∗ *root*, **rb_tree** ∗ *z*)

4.48.1.8 void rb_delete_fixup (**rb_tree** ∗∗ *root*, **rb_tree** ∗ *x*)

4.48.1.9 void∗ rb_delete_key (**rb_tree** ∗∗ *root*, void ∗ *key*, int *ikey*, int *compareint* ∗*, int* ∗)

4.48.1.10 void∗ rb_find (**rb_tree** ∗ *root*, void ∗ *key*, int *ikey*, int *compareint* ∗*, int* ∗)

4.48.1.11 void rb_get_sorted_order (**rb_tree** ∗ *root*, int *compareint* ∗*, int* ∗*,* **llist** ∗ *sorted*)

4.48.1.12 void rb_get_sorted_order_list (**rb_tree** ∗ *root*, int *compareint* ∗*, int* ∗*,* **llist** ∗ *sorted*, **llist** ∗ *keys*)

4.48.1.13 void∗ rb_insert (**rb_tree** ∗∗ *root*, void ∗ *key*, int *ikey*, void ∗ *payload*, int *compareint* ∗*, int* ∗)

4.48.1.14 void rb_insert_fixup (**rb_tree** ∗∗ *root*, **rb_tree** ∗ *z*)

4.48.1.15 void rb_left_rotate (**rb_tree** ∗∗ *root*, **rb_tree** ∗ *x*)

4.48.1.16 **rb_tree**∗ rb_maximum (**rb_tree** ∗ *x*)

4.48.1.17 **rb_tree**∗ rb_minimum (**rb_tree** ∗ *x*)

4.48.1.18 void rb_print_tree (**rb_tree** ∗ *root*)

4.48.1.19 void rb_print_tree_l (**rb_tree** ∗ *root*, int *l*)

4.48.1.20 void rb_right_rotate (**rb_tree** ∗∗ *root*, **rb_tree** ∗ *x*)

4.48.1.21 **rb_tree**∗ rb_successor (**rb_tree** ∗ *x*)

4.48.1.22 **rb_tree**∗ rb_tree_union (**rb_tree** ∗ *a*, **rb_tree** ∗ *b*, int *compareint* ∗*, int* ∗)

4.49 rb_tree.h File Reference

```
#include "llist.h"
#include "utils.h"
```

Classes

- struct rb_tree

Typedefs

- typedef struct rb_tree rb_tree

Functions

- void destroy_rb_tree (rb_tree ∗∗root)
- void destroy_rb_tree_key (rb_tree ∗∗root, void destroyer(llist ∗))
- void destroy_rb_tree_paload (rb_tree ∗∗root, void destroyer(llist ∗))
- rb_tree ∗ new_rb_tree (void ∗key, int ikey, void ∗payload)
- void rb_left_rotate (rb_tree ∗∗root, rb_tree ∗x)
- void rb_right_rotate (rb_tree ∗∗root, rb_tree ∗x)
- void ∗ rb_insert (rb_tree ∗∗root, void ∗key, int ikey, void ∗payload, int compare(int ∗, int ∗))
- void rb_insert_fixup (rb_tree ∗∗root, rb_tree ∗z)
- void ∗ rb_find (rb_tree ∗root, void ∗key, int ikey, int compare(int ∗, int ∗))
- void rb_get_sorted_order (rb_tree ∗root, int compare(int ∗, int ∗), llist ∗sorted)
- void rb_get_sorted_order_list (rb_tree ∗root, int compare(int ∗, int ∗), llist ∗sorted, llist ∗keys)
- rb_tree ∗ rb_copy_tree (rb_tree ∗tree, int compare(int ∗, int ∗))
- rb_tree ∗ rb_successor (rb_tree ∗x)
- rb_tree ∗ rb_minimum (rb_tree ∗x)
- rb_tree ∗ rb_maximum (rb_tree ∗x)
- void rb_delete_fixup (rb_tree ∗∗root, rb_tree ∗x)
- void ∗ rb_delete (rb_tree ∗∗root, rb_tree ∗z)
- void ∗ rb_delete_key (rb_tree ∗∗root, void ∗key, int ikey, int compare(int ∗, int ∗))
- void rb_print_tree_l (rb_tree ∗root, int l)
- void rb_print_tree (rb_tree ∗root)
- void check_tree (rb_tree ∗root)
- rb_tree ∗ rb_tree_union (rb_tree ∗a, rb_tree ∗b, int compare(int ∗, int ∗))

4.49.1 Detailed Description

Provides a red-black tree implementation.

4.49.2 Typedef Documentation

4.49.2.1 typedef struct rb_tree rb_tree

4.49.3 Function Documentation

4.49.3.1 void check_tree (rb_tree ∗ *root*)

4.49.3.2 void destroy_rb_tree (rb_tree ∗∗ *root*)

4.49.3.3 void destroy_rb_tree_key (rb_tree ∗∗ *root,* void *destroyerllist* ∗)

4.49.3.4 void destroy_rb_tree_paload (rb_tree ∗∗ *root,* void *destroyerllist* ∗)

4.49.3.5 rb_tree∗ new_rb_tree (void ∗ *key,* int *ikey,* void ∗ *payload*)

4.49.3.6 rb_tree∗ rb_copy_tree (rb_tree ∗ *tree,* int *compareint* ∗*, int* ∗)

4.49.3.7 void∗ rb_delete (rb_tree ∗∗ *root,* rb_tree ∗ *z*)

4.49.3.8 void rb_delete_fixup (rb_tree ∗∗ *root,* rb_tree ∗ *x*)

4.49.3.9 void∗ rb_delete_key (rb_tree ∗∗ *root,* void ∗ *key,* int *ikey,* int *compareint* ∗*, int* ∗)

4.49.3.10 void∗ rb_find (rb_tree ∗ *root,* void ∗ *key,* int *ikey,* int *compareint* ∗*, int* ∗)

4.49.3.11 void rb_get_sorted_order (rb_tree ∗ *root,* int *compareint* ∗*, int* ∗*,* llist ∗ *sorted*)

4.49.3.12 void rb_get_sorted_order_list (rb_tree ∗ *root,* int *compareint* ∗*, int* ∗*,* llist ∗ *sorted,* llist ∗ *keys*)

4.49.3.13 void∗ rb_insert (rb_tree ∗∗ *root,* void ∗ *key,* int *ikey,* void ∗ *payload,* int *compareint* ∗*, int* ∗)

4.49.3.14 void rb_insert_fixup (rb_tree ∗∗ *root,* rb_tree ∗ *z*)

4.49.3.15 void rb_left_rotate (rb_tree ∗∗ *root,* rb_tree ∗ *x*)

4.49.3.16 rb_tree∗ rb_maximum (rb_tree ∗ *x*)

4.49.3.17 rb_tree∗ rb_minimum (rb_tree ∗ *x*)

4.49.3.18 void rb_print_tree (rb_tree ∗ *root*)

4.49.3.19 void rb_print_tree_l (rb_tree ∗ *root,* int *l*)

4.49.3.20 void rb_right_rotate (rb_tree ∗∗ *root,* rb_tree ∗ *x*)

4.49.3.21 rb_tree∗ rb_successor (rb_tree ∗ *x*)

4.49.3.22 rb_tree∗ rb_tree_union (rb_tree ∗ *a,* rb_tree ∗ *b,* int *compareint* ∗*, int* ∗)

4.50 rbt_set.c File Reference

```
#include "rbt_set.h"
#include "utils.h"
```

Functions

- rbt_set ∗ new_rbt_set (int size)
- int add_element_rbt (rbt_set ∗s, void ∗e, int key)
- int set_empty_rbt (rbt_set ∗s)
- void destroy_set_rbt (rbt_set ∗s)
- void ∗ remove_element_with_key_rbt (rbt_set ∗s, int key)

- void ∗ get_element_with_key_rbt (rbt_set ∗s, int key)
- int remove_int_element_rbt (rbt_set ∗s, int key)
- int get_int_element_rbt (rbt_set ∗s, int key)
- int add_int_element_rbt (rbt_set ∗s, int key)
- rbt_set ∗ copy_set_rbt (rbt_set ∗s)
- void reset_set_rbt (rbt_set ∗s)
- llist ∗ set_to_list_rbt (rbt_set ∗s)

4.50.1 Function Documentation

4.50.1.1 int add_element_rbt (**rbt_set** ∗ ***s,*** void ∗ ***e,*** int ***key***)

4.50.1.2 int add_int_element_rbt (**rbt_set** ∗ ***s,*** int ***key***)

4.50.1.3 **rbt_set**∗ copy_set_rbt (**rbt_set** ∗ ***s***)

4.50.1.4 void destroy_set_rbt (**rbt_set** ∗ ***s***)

4.50.1.5 void∗ get_element_with_key_rbt (**rbt_set** ∗ ***s,*** int ***key***)

4.50.1.6 int get_int_element_rbt (**rbt_set** ∗ ***s,*** int ***key***)

4.50.1.7 **rbt_set**∗ new_rbt_set (int ***size***)

4.50.1.8 void∗ remove_element_with_key_rbt (**rbt_set** ∗ ***s,*** int ***key***)

4.50.1.9 int remove_int_element_rbt (**rbt_set** ∗ ***s,*** int ***key***)

4.50.1.10 void reset_set_rbt (**rbt_set** ∗ ***s***)

4.50.1.11 int set_empty_rbt (**rbt_set** ∗ ***s***)

4.50.1.12 **llist**∗ set_to_list_rbt (**rbt_set** ∗ ***s***)

4.51 rbt_set.h File Reference

```
#include "set.h"
#include "rb_tree.h"
```

Classes

- struct rbt_set

Typedefs

- typedef struct rbt_set rbt_set

Functions

- rbt_set ∗ new_rbt_set (int size)
- int add_element_rbt (rbt_set ∗s, void ∗e, int key)
- int set_empty_rbt (rbt_set ∗s)

- void destroy_set_rbt (rbt_set ∗s)
- void ∗ peek_element_rbt (rbt_set ∗s)
- void ∗ remove_element_with_key_rbt (rbt_set ∗s, int key)
- void ∗ get_element_with_key_rbt (rbt_set ∗s, int key)
- int remove_int_element_rbt (rbt_set ∗s, int key)
- int get_int_element_rbt (rbt_set ∗s, int key)
- int add_int_element_rbt (rbt_set ∗s, int key)
- rbt_set ∗ copy_set_rbt (rbt_set ∗s)
- void reset_set_rbt (rbt_set ∗s)
- llist ∗ set_to_list_rbt (rbt_set ∗s)

4.51.1 Detailed Description

Provides a set implementation using red-black trees.

4.51.2 Typedef Documentation

4.51.2.1 typedef struct rbt_set rbt_set

4.51.3 Function Documentation

4.51.3.1 int add_element_rbt (**rbt_set** ∗ *s,* void ∗ *e,* int *key*)

4.51.3.2 int add_int_element_rbt (**rbt_set** ∗ *s,* int *key*)

4.51.3.3 **rbt_set**∗ copy_set_rbt (**rbt_set** ∗ *s*)

4.51.3.4 void destroy_set_rbt (**rbt_set** ∗ *s*)

4.51.3.5 void∗ get_element_with_key_rbt (**rbt_set** ∗ *s,* int *key*)

4.51.3.6 int get_int_element_rbt (**rbt_set** ∗ *s,* int *key*)

4.51.3.7 **rbt_set**∗ new_rbt_set (int *size*)

4.51.3.8 void∗ peek_element_rbt (**rbt_set** ∗ *s*)

4.51.3.9 void∗ remove_element_with_key_rbt (**rbt_set** ∗ *s,* int *key*)

4.51.3.10 int remove_int_element_rbt (**rbt_set** ∗ *s,* int *key*)

4.51.3.11 void reset_set_rbt (**rbt_set** ∗ *s*)

4.51.3.12 int set_empty_rbt (**rbt_set** ∗ *s*)

4.51.3.13 **llist**∗ set_to_list_rbt (**rbt_set** ∗ *s*)

4.52 rpwc.c File Reference

```
#include "rpwc.h"
#include "utils.h"
```

Functions

- lastGAC ∗ new_lastGAC (htable ∗ht, int varid)
- void destroy_lastGAC (lastGAC ∗lg)
- undo_lastGAC ∗ new_undo_lastGAC (void)
- undo_info ∗ new_undo_info (void)
- void destroy_undo_info (undo_info ∗ui)
- void init_rpwc (variables ∗vars)
- tripplet ∗ new_tripplet (int a, int b, int c)
- int reviseMaxRPWC (variable ∗var, constraint ∗cons, llist ∗∗reduct, undo_lastGAC ∗∗undo_LG)
- int reviseGAC (variable ∗var, constraint ∗cons, llist ∗∗reduct, undo_lastGAC ∗∗undo_LG)
- int reviseGACMaxRPWC (variable ∗var, constraint ∗cons, llist ∗∗reduct, undo_lastGAC ∗∗undo_LG, int algo)
- int domain_filter (variable ∗c_var)
- void enqueue (variable ∗var, constraint ∗ci, set ∗queue)

4.52.1 Function Documentation

4.52.1.1 void destroy_lastGAC (**lastGAC** ∗ *lg*)

4.52.1.2 void destroy_undo_info (**undo_info** ∗ *ui*)

4.52.1.3 int domain_filter (**variable** ∗ *c_var*)

4.52.1.4 void enqueue (**variable** ∗ *var,* **constraint** ∗ *ci,* **set** ∗ *queue*)

4.52.1.5 void init_rpwc (**variables** ∗ *vars*)

4.52.1.6 **lastGAC**∗ new_lastGAC (**htable** ∗ *ht,* int *varid*)

4.52.1.7 **tripplet**∗ new_tripplet (int *a,* int *b,* int *c*)

4.52.1.8 **undo_info**∗ new_undo_info (void)

4.52.1.9 **undo_lastGAC**∗ new_undo_lastGAC (void)

4.52.1.10 int reviseGAC (**variable** ∗ *var,* **constraint** ∗ *cons,* **llist** ∗∗ *reduct,* **undo_lastGAC** ∗∗ *undo_LG*)

4.52.1.11 int reviseGACMaxRPWC (**variable** ∗ *var,* **constraint** ∗ *cons,* **llist** ∗∗ *reduct,* **undo_lastGAC** ∗∗ *undo_LG,* int *algo*)

4.52.1.12 int reviseMaxRPWC (**variable** ∗ *var,* **constraint** ∗ *cons,* **llist** ∗∗ *reduct,* **undo_lastGAC** ∗∗ *undo_LG*)

4.53 rpwc.h File Reference

```
#include "variables.h"
#include "hashtable.h"
#include "llist.h"
#include "domains.h"
#include "globals.h"
#include "query_graph.h"
#include "set.h"
#include "utils.h"
```

Classes

- struct lastGAC
- struct undo_lastGAC
- struct tripplet
- struct undo_info

Typedefs

- typedef struct lastGAC lastGAC
- typedef struct undo_lastGAC undo_lastGAC
- typedef struct tripplet tripplet
- typedef struct undo_info undo_info

Enumerations

- enum reviseAlgo { NONE, GAC, maxRPWC }

Functions

- undo_lastGAC ∗ new_undo_lastGAC (void)
- tripplet ∗ new_tripplet (int a, int b, int c)
- undo_info ∗ new_undo_info (void)
- void destroy_undo_info (undo_info ∗ui)
- lastGAC ∗ new_lastGAC (htable ∗ht, int varid)
- void destroy_lastGAC (lastGAC ∗lg)
- void init_rpwc (variables ∗vars)
- int reviseGACMaxRPWC (variable ∗var, constraint ∗cons, llist ∗∗reduct, undo_lastGAC ∗∗undo_LG, int algo)
- int reviseMaxRPWC (variable ∗var, constraint ∗cons, llist ∗∗reduct, undo_lastGAC ∗∗undo_LG)
- int reviseGAC (variable ∗var, constraint ∗cons, llist ∗∗reduct, undo_lastGAC ∗∗undo_LG)
- int domain_filter (variable ∗c_var)
- void enqueue (variable ∗var, constraint ∗ci, set ∗queue)

4.53.1 Detailed Description

Provides the implementations for the GAC and maxRPWC algorithms.

4.53.2 Typedef Documentation

4.53.2.1 typedef struct lastGAC lastGAC

4.53.2.2 typedef struct tripplet tripplet

4.53.2.3 typedef struct undo_info undo_info

4.53.2.4 typedef struct undo_lastGAC undo_lastGAC

4.53.3 Enumeration Type Documentation

4.53.3.1 enum reviseAlgo

Enumerator

NONE

GAC

maxRPWC

4.53.4 Function Documentation

4.53.4.1 void destroy_lastGAC (**lastGAC** ∗ *lg*)

4.53.4.2 void destroy_undo_info (**undo_info** ∗ *ui*)

4.53.4.3 int domain_filter (**variable** ∗ *c_var*)

4.53.4.4 void enqueue (**variable** ∗ *var*, **constraint** ∗ *ci*, **set** ∗ *queue*)

4.53.4.5 void init_rpwc (**variables** ∗ *vars*)

4.53.4.6 **lastGAC**∗ new_lastGAC (**htable** ∗ *ht*, int *varid*)

4.53.4.7 **tripplet**∗ new_tripplet (int *a*, int *b*, int *c*)

4.53.4.8 **undo_info**∗ new_undo_info (void)

4.53.4.9 **undo_lastGAC**∗ new_undo_lastGAC (void)

4.53.4.10 int reviseGAC (**variable** ∗ *var*, **constraint** ∗ *cons*, **llist** ∗∗ *reduct*, **undo_lastGAC** ∗∗ *undo_LG*)

4.53.4.11 int reviseGACMaxRPWC (**variable** ∗ *var*, **constraint** ∗ *cons*, **llist** ∗∗ *reduct*, **undo_lastGAC** ∗∗ *undo_LG*, int *algo*)

4.53.4.12 int reviseMaxRPWC (**variable** ∗ *var*, **constraint** ∗ *cons*, **llist** ∗∗ *reduct*, **undo_lastGAC** ∗∗ *undo_LG*)

4.54 rr.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include "utils.h"
#include "llist.h"
#include "set.h"
#include "rr.h"
#include "rb_tree.h"
#include "tables.h"
```

Functions

- constraint_graph ∗ rr (constraint_graph ∗cg, int cons_last_id)
- rb_tree ∗ get_overlaps (constraint_graph ∗cg)
- llist ∗ get_relation_groups (llist ∗sorted_overlaps, llist ∗sorted_overlaps_keys)
- void add_rr_edges (llist ∗connected_components, constraint_graph ∗g_sub_k)

4.54.1 Function Documentation

4.54.1.1 void add_rr_edges (**llist** ∗ *connected_components*, **constraint_graph** ∗ *g_sub_k*)

4.54.1.2 rb_tree∗ get_overlaps (constraint_graph ∗ *cg*)

4.54.1.3 llist∗ get_relation_groups (llist ∗ *sorted_overlaps,* llist ∗ *sorted_overlaps_keys*)

4.54.1.4 constraint_graph∗ rr (constraint_graph ∗ *cg,* int *cons_last_id*)

4.55 rr.h File Reference

```
#include "query_graph.h"
#include "constraints.h"
#include "rb_tree.h"
```

Functions

- constraint_graph ∗ rr (constraint_graph ∗cg, int cons_last_id)
- rb_tree ∗ get_overlaps (constraint_graph ∗cg)
- llist ∗ get_relation_groups (llist ∗overlaps, llist ∗sorted_overlaps_keys)
- void add_rr_edges (llist ∗connected_components, constraint_graph ∗g_sub_k)

4.55.1 Detailed Description

Implements the redundancy-removal algorithm.

4.55.2 Function Documentation

4.55.2.1 void add_rr_edges (llist ∗ *connected_components,* constraint_graph ∗ *g_sub_k*)

4.55.2.2 rb_tree∗ get_overlaps (constraint_graph ∗ *cg*)

4.55.2.3 llist∗ get_relation_groups (llist ∗ *overlaps,* llist ∗ *sorted_overlaps_keys*)

4.55.2.4 constraint_graph∗ rr (constraint_graph ∗ *cg,* int *cons_last_id*)

4.56 rs2.c File Reference

```
#include "rs2.h"
#include "tree.h"
#include "utils.h"
#include "llist.h"
#include "tree_map.h"
#include "query_graph.h"
#include "rs2tree.h"
#include "oneall.h"
#include "gbtd.h"
```

Functions

- int stage4_filter_with_no_check_insol_tuples (set ∗queue, int time, set ∗undo_set, set ∗cluster_vars, light_stack ∗undo_stack)

- int stage4_filter_with_all_solutions (set ∗queue, int time, set ∗undo_set, set ∗cluster_vars, light_stack ∗undo_stack, int ∗queues_total_count, int ppr)
- int stage4_filter_auto (set ∗queue, set ∗∗queues, int time, set ∗undo_set, set ∗cluster_vars, light_stack ∗undo_stack, int ∗queues_total_count, int ppr)
- void get_rstat (comb_data ∗cdata)
- int stage4_filter_auto_one_sol (set ∗queue, set ∗∗queues, int time, set ∗undo_set, light_stack ∗undo_stack, int ∗queues_total_count)
- int stage4_filter_auto_all_sol (set ∗queue, set ∗∗queues, int time, set ∗undo_set, light_stack ∗undo_stack, int ∗queues_total_count)
- int stage4_filter (set ∗queue, int time, set ∗undo_set, set ∗cluster_vars, light_stack ∗undo_stack, int ppr)
- int stage4_filter_multiqueue (set ∗queue, int time, set ∗undo_set, set ∗cluster_vars, light_stack ∗undo_stack, set ∗∗queues, int ∗queues_total_count, int ppr)
- set ∗ stage4_init_queue (llist ∗constraint_graph_node_list)
- void queue_relation_related (htable ∗ht, set ∗queue)
- void queue_relation_related_multiqueue (htable ∗ht, set ∗queue, set ∗∗queues, int ∗queues_total_count)
- int setup_relation_groups (llist ∗combs, int context, options category)
- int setup_relation_groups_ (llist ∗combs, int context, options category, comb_data ∗∗cdatas, llist ∗context_list)
- void enum_table_pairs (htable ∗hta0, llist_node ∗combnp, int ∗varmap, comb_data ∗cd, comb_data ∗base_cd)
- comb_data ∗ make_comb_data2 (llist ∗comb, llist_node ∗combnp, int context, int no_tuple_positions, options category, comb_data ∗base_cd)
- htable ∗ get_hts_paired (comb_data ∗cd, int hi, int hj, int p)
- tree_map ∗ get_tms_paired (comb_data ∗cd, int hi, int hj)
- llist ∗ get_pair_common_cols (comb_data ∗cd, int hi, int hj)
- void destroy_comb_data2 (comb_data ∗cd)
- void build_ind_trees (comb_data ∗cd, llist_node ∗combnp, int ∗varmap, comb_data ∗base_cd)
- int find_support_for_single_solution (int t, comb_data ∗cd, set ∗cluster_vars, double ∗sol_count)
- int get_revision_time (htable ∗ht, int t)
- void set_revision_time (htable ∗ht, int t, int revision_time, comb_data ∗cd)
- void set_revision_time_parents (htable ∗ht, int t, int revision_time, comb_data ∗cd, llist ∗stack, int ∗∗tuples)
- int find_support_for_single_solution_bc (int t, comb_data ∗cd)
- int find_support_for_single_solution_bc_rec (int ∗∗tuples, comb_data ∗cd, int level, int ∗tuple_positions, int ∗tuple_position_htid)
- int find_support_for_single_solution_with_fc (int t, comb_data ∗cd, int first_table, int all_solutions, llist ∗remaining_vars, set ∗cluster_vars, active_tuple ∗ac, double ∗sol_count)
- int find_support_for_single_solution_with_fc_priority (int t, comb_data ∗cd, int first_table, int all_solutions, llist ∗remaining_vars, set ∗cluster_vars, active_tuple ∗ac, int priority_last_n, double ∗sol_count)
- htable ∗ get_ht_from_cd (comb_data ∗cd, int ht_pos)
- int mark_all_in_future (int ∗old_marks, int mark_with, int var, llist ∗remaining_vars, int ∗current_tuple, comb_data ∗cd, llist ∗undo_dom, int ∗marks, llist ∗∗∗doms, int rec_depth, int drop_for_all_revised)
- void set_deg_count (comb_data ∗cd, llist ∗remaining_vars, llist ∗undo_deg, int removed)
- int choose_var (comb_data ∗cd, llist ∗remaining_vars, llist ∗undo_deg)
- int choose_var_priority (comb_data ∗cd, llist ∗remaining_vars, llist ∗undo_deg, int priority_last_n)
- int choose_var_priority_decided (comb_data ∗cd, llist ∗remaining_vars, llist ∗undo_deg, int priority_last_n, int decided, active_tuple ∗ac)
- void set_dom (comb_data ∗cd)
- void undo_dom_deg_var (llist ∗undo_deg, llist ∗undo_dom, comb_data ∗cd, int var, llist ∗remaining_vars, llist ∗∗doms, llist ∗∗doms_xlevel)
- int visit_rest_of_tuples (llist ∗remaining_vars, int all_solutions, int next_var, comb_data ∗cd, int var, int ∗old_marks, int ∗∗tuples, llist ∗∗∗doms, int rec_depth, int ∗tuple_positions, llist ∗undo_deg, llist ∗undo_dom)
- int find_support_for_single_solution_with_fc_rec_single_out (int rec_depth, int ∗∗allocated_marks, int ∗old_marks, int ∗∗tuples, comb_data ∗cd, int var, int ∗tuple_positions, int ∗tuple_position_htid, llist ∗remaining_vars, int all_solutions, llist ∗∗∗doms, set ∗cluster_vars, active_tuple ∗ac, int priority_last_n, double ∗sol_count, llist ∗stack)

- int find_support_for_single_solution_with_fc_rec (int rec_depth, int ∗∗allocated_marks, int ∗old_marks, int ∗∗tuples, comb_data ∗cd, int var, int ∗tuple_positions, int ∗tuple_position_htid, llist ∗remaining_vars, int all_solutions, llist ∗∗∗doms, set ∗cluster_vars, active_tuple ∗ac, int priority_last_n, double ∗sol_count, llist ∗stack)
- int filter_all_relations_with_all_solutions (comb_data ∗cd, set ∗cluster_vars, active_tuple ∗ac, double ∗sol_count)
- int filter_all_relations_with_all_solutions_priority (comb_data ∗cd, set ∗cluster_vars, active_tuple ∗ac, int priority_last_n, double ∗sol_count)
- void find_all_supports_with_all_solutions_with_fc (comb_data ∗cd, set ∗cluster_vars, active_tuple ∗ac, double ∗sol_count)
- void find_all_supports_with_all_solutions_with_fc_priority (comb_data ∗cd, set ∗cluster_vars, active_tuple ∗ac, int priority_last_n, double ∗sol_count)
- void find_all_supports_with_all_solutions_with_fc_priority_next_var (comb_data ∗cd, set ∗cluster_vars, active_tuple ∗ac, int priority_last_n, int next_v, double ∗sol_count)
- int filter_single_relation_in_combination (comb_data ∗cd, set ∗undo_set, light_stack ∗undo_stack, double ∗sol_count)
- comb_data ∗ setup_relation_group_for_a_comb (llist ∗comb, int context, options category, comb_data ∗base_cd)
- comb_data ∗ setup_relation_group_for_a_comb_first (llist ∗comb, int context, llist_node ∗firsttable, options category, comb_data ∗base_cd)
- comb_data ∗ setup_relation_group_for_a_comb_long_indexes (llist ∗comb, options category)
- llist ∗ get_comb_from (htable ∗filter, htable ∗given)
- void destroy_comb (llist ∗comb)
- comb_cdata ∗ new_comb_cdata (int context)
- void destroy_comb_cdata (comb_cdata ∗ccd)
- void clear_comb_datas_from_hts (llist ∗htlist)

Variables

- void ∗ watch = NULL

4.56.1 Function Documentation

4.56.1.1 void build_ind_trees (comb_data ∗ *cd,* llist_node ∗ *combnp,* int ∗ *varmap,* comb_data ∗ *base_cd*)

4.56.1.2 int choose_var (comb_data ∗ *cd,* llist ∗ *remaining_vars,* llist ∗ *undo_deg*)

4.56.1.3 int choose_var_priority (comb_data ∗ *cd,* llist ∗ *remaining_vars,* llist ∗ *undo_deg,* int *priority_last_n*)

4.56.1.4 int choose_var_priority_decided (comb_data ∗ *cd,* llist ∗ *remaining_vars,* llist ∗ *undo_deg,* int *priority_last_n,* int *decided,* active_tuple ∗ *ac*)

4.56.1.5 void clear_comb_datas_from_hts (llist ∗ *htlist*)

4.56.1.6 void destroy_comb (llist ∗ *comb*)

4.56.1.7 void destroy_comb_cdata (comb_cdata ∗ *ccd*)

4.56.1.8 void destroy_comb_data2 (comb_data ∗ *cd*)

4.56.1.9 void enum_table_pairs (htable ∗ *hta0,* llist_node ∗ *combnp,* int ∗ *varmap,* comb_data ∗ *cd,* comb_data ∗ *base_cd*)

4.56.1.10 int filter_all_relations_with_all_solutions (comb_data ∗ *cd,* set ∗ *cluster_vars,* active_tuple ∗ *ac,* double ∗ *sol_count*)

4.56.1.11 int filter_all_relations_with_all_solutions_priority (comb_data * cd, set * cluster_vars, active_tuple * ac, int priority_last_n, double * sol_count)

4.56.1.12 int filter_single_relation_in_combination (comb_data * cd, set * undo_set, light_stack * undo_stack, double * sol_count)

4.56.1.13 void find_all_supports_with_all_solutions_with_fc (comb_data * cd, set * cluster_vars, active_tuple * ac, double * sol_count)

4.56.1.14 void find_all_supports_with_all_solutions_with_fc_priority (comb_data * cd, set * cluster_vars, active_tuple * ac, int priority_last_n, double * sol_count)

4.56.1.15 void find_all_supports_with_all_solutions_with_fc_priority_next_var (comb_data * cd, set * cluster_vars, active_tuple * ac, int priority_last_n, int next_v, double * sol_count)

4.56.1.16 int find_support_for_single_solution (int t, comb_data * cd, set * cluster_vars, double * sol_count)

4.56.1.17 int find_support_for_single_solution_bc (int t, comb_data * cd)

4.56.1.18 int find_support_for_single_solution_bc_rec (int ** tuples, comb_data * cd, int level, int * tuple_positions, int * tuple_position_htid)

4.56.1.19 int find_support_for_single_solution_with_fc (int t, comb_data * cd, int first_table, int all_solutions, llist * remaining_vars, set * cluster_vars, active_tuple * ac, double * sol_count)

4.56.1.20 int find_support_for_single_solution_with_fc_priority (int t, comb_data * cd, int first_table, int all_solutions, llist * remaining_vars, set * cluster_vars, active_tuple * ac, int priority_last_n, double * sol_count)

4.56.1.21 int find_support_for_single_solution_with_fc_rec (int rec_depth, int ** allocated_marks, int * old_marks, int ** tuples, comb_data * cd, int var, int * tuple_positions, int * tuple_position_htid, llist * remaining_vars, int all_solutions, llist *** doms, set * cluster_vars, active_tuple * ac, int priority_last_n, double * sol_count, llist * stack)

4.56.1.22 int find_support_for_single_solution_with_fc_rec_single_out (int rec_depth, int ** allocated_marks, int * old_marks, int ** tuples, comb_data * cd, int var, int * tuple_positions, int * tuple_position_htid, llist * remaining_vars, int all_solutions, llist *** doms, set * cluster_vars, active_tuple * ac, int priority_last_n, double * sol_count, llist * stack)

4.56.1.23 llist* get_comb_from (htable * filter, htable * given)

4.56.1.24 htable* get_ht_from_cd (comb_data * cd, int ht_pos)

4.56.1.25 htable* get_hts_paired (comb_data * cd, int hi, int hj, int p)

4.56.1.26 llist* get_pair_common_cols (comb_data * cd, int hi, int hj)

4.56.1.27 int get_revision_time (htable * ht, int t)

4.56.1.28 void get_rstat (comb_data * cdata)

4.56.1.29 tree_map* get_tms_paired (comb_data * cd, int hi, int hj)

4.56.1.30 comb_data* make_comb_data2 (llist * comb, llist_node * combnp, int context, int no_tuple_positions, options category, comb_data * base_cd)

4.56.1.31 int mark_all_in_future (int ∗ *old_marks,* int *mark_with,* int *var,* llist ∗ *remaining_vars,* int ∗ *current_tuple,* comb_data ∗ *cd,* llist ∗ *undo_dom,* int ∗ *marks,* llist ∗∗∗ *doms,* int *rec_depth,* int *drop_for_all_revised*)

4.56.1.32 comb_cdata∗ new_comb_cdata (int *context*)

4.56.1.33 void queue_relation_related (htable ∗ *ht,* set ∗ *queue*)

4.56.1.34 void queue_relation_related_multiqueue (htable ∗ *ht,* set ∗ *queue,* set ∗∗ *queues,* int ∗ *queues_total_count*)

4.56.1.35 void set_deg_count (comb_data ∗ *cd,* llist ∗ *remaining_vars,* llist ∗ *undo_deg,* int *removed*)

4.56.1.36 void set_dom (comb_data ∗ *cd*)

4.56.1.37 void set_revision_time (htable ∗ *ht,* int *t,* int *revision_time,* comb_data ∗ *cd*)

4.56.1.38 void set_revision_time_parents (htable ∗ *ht,* int *t,* int *revision_time,* comb_data ∗ *cd,* llist ∗ *stack,* int ∗∗ *tuples*)

4.56.1.39 comb_data∗ setup_relation_group_for_a_comb (llist ∗ *comb,* int *context,* options *category,* comb_data ∗ *base_cd*)

4.56.1.40 comb_data∗ setup_relation_group_for_a_comb_first (llist ∗ *comb,* int *context,* llist_node ∗ *firsttable,* options *category,* comb_data ∗ *base_cd*)

4.56.1.41 comb_data∗ setup_relation_group_for_a_comb_long_indexes (llist ∗ *comb,* options *category*)

4.56.1.42 int setup_relation_groups (llist ∗ *combs,* int *context,* options *category*)

4.56.1.43 int setup_relation_groups_ (llist ∗ *combs,* int *context,* options *category,* comb_data ∗∗ *cdatas,* llist ∗ *context_list*)

4.56.1.44 int stage4_filter (set ∗ *queue,* int *time,* set ∗ *undo_set,* set ∗ *cluster_vars,* light_stack ∗ *undo_stack,* int *ppr*)

4.56.1.45 int stage4_filter_auto (set ∗ *queue,* set ∗∗ *queues,* int *time,* set ∗ *undo_set,* set ∗ *cluster_vars,* light_stack ∗ *undo_stack,* int ∗ *queues_total_count,* int *ppr*)

4.56.1.46 int stage4_filter_auto_all_sol (set ∗ *queue,* set ∗∗ *queues,* int *time,* set ∗ *undo_set,* light_stack ∗ *undo_stack,* int ∗ *queues_total_count*)

4.56.1.47 int stage4_filter_auto_one_sol (set ∗ *queue,* set ∗∗ *queues,* int *time,* set ∗ *undo_set,* light_stack ∗ *undo_stack,* int ∗ *queues_total_count*)

4.56.1.48 int stage4_filter_multiqueue (set ∗ *queue,* int *time,* set ∗ *undo_set,* set ∗ *cluster_vars,* light_stack ∗ *undo_stack,* set ∗∗ *queues,* int ∗ *queues_total_count,* int *ppr*)

4.56.1.49 int stage4_filter_with_all_solutions (set ∗ *queue,* int *time,* set ∗ *undo_set,* set ∗ *cluster_vars,* light_stack ∗ *undo_stack,* int ∗ *queues_total_count,* int *ppr*)

4.56.1.50 int stage4_filter_with_no_check_insol_tuples (set ∗ *queue,* int *time,* set ∗ *undo_set,* set ∗ *cluster_vars,* light_stack ∗ *undo_stack*)

4.56.1.51 set∗ stage4_init_queue (llist ∗ *constraint_graph_node_list*)

4.56.1.52 void undo_dom_deg_var (llist ∗ *undo_deg,* llist ∗ *undo_dom,* comb_data ∗ *cd,* int *var,* llist ∗ *remaining_vars,* llist ∗∗ *doms,* llist ∗∗ *doms_xlevel*)

4.56.1.53 int visit_rest_of_tuples (llist ∗ *remaining_vars,* int *all_solutions,* int *next_var,* comb_data ∗ *cd,* int *var,* int ∗ *old_marks,* int ∗∗ *tuples,* llist ∗∗∗ *doms,* int *rec_depth,* int ∗ *tuple_positions,* llist ∗ *undo_deg,* llist ∗ *undo_dom*)

4.56.2 Variable Documentation

4.56.2.1 void∗ watch = NULL

4.57 rs2.h File Reference

```
#include "query_graph.h"
#include "llist.h"
#include "hashtable.h"
#include "set.h"
#include "tree_map.h"
#include "rs.h"
#include "active_tuple.h"
```

Classes

- struct table_map
- struct comb_cdata

Typedefs

- typedef struct table_map table_map
- typedef struct comb_cdata comb_cdata

Functions

- htable ∗ get_hts_paired (comb_data ∗cd, int hi, int hj, int p)
- tree_map ∗ get_tms_paired (comb_data ∗cd, int hi, int hj)
- llist ∗ get_pair_common_cols (comb_data ∗cd, int hi, int hj)
- comb_data ∗ make_comb_data2 (llist ∗comb, llist_node ∗combnp, int context, int no_tuple_positions, options category, comb_data ∗base_cd)
- void destroy_comb_data2 (comb_data ∗cd)
- void filter_relations_rs (llist ∗queue)
- void build_joining_order (htable ∗hta, llist_node ∗combnp, int ∗vmap, llist ∗hts_ordered, llist ∗all_com_cols)
- void build_ind_trees (comb_data ∗cd, llist_node ∗combnp, int ∗vmap, comb_data ∗base_cd)
- int find_support_for_single_solution (int t, comb_data ∗cd, set ∗cluster_vars, double ∗sol_count)
- int find_support_for_single_solution_rec (int ∗∗tuples, comb_data ∗cd, int level, int ∗tuple_positions, int ∗tuple_position_htid)
- int setup_relation_groups (llist ∗combs, int context, options category)
- int setup_relation_groups_ (llist ∗combs, int context, options category, comb_data ∗∗cdatas, llist ∗context_-list)
- set ∗ stage4_init_queue (llist ∗constraint_graph_node_list)
- int stage4_filter_auto (set ∗queue, set ∗∗queues, int time, set ∗undo_set, set ∗cluster_vars, light_stack ∗undo_stack, int ∗queues_total_count, int ppr)
- int stage4_filter_auto_one_sol (set ∗queue, set ∗∗queues, int time, set ∗undo_set, light_stack ∗undo_stack, int ∗queues_total_count)
- int stage4_filter_auto_all_sol (set ∗queue, set ∗∗queues, int time, set ∗undo_set, light_stack ∗undo_stack, int ∗queues_total_count)
- int stage4_filter (set ∗queue, int time, set ∗undo_set, set ∗cluster_vars, light_stack ∗undo_stack, int ppr)
- int stage4_filter_multiqueue (set ∗queue, int time, set ∗undo_set, set ∗cluster_vars, light_stack ∗undo_stack, set ∗∗queues, int ∗queues_total_count, int ppr)
- void queue_related_relations (set ∗queue, htable ∗ht)

- support ∗ new_support (comb_data ∗comb, int tuple)
- void destroy_support (support ∗s)
- update_support ∗ new_update_support (comb_data ∗comb)
- void queue_supported_tuples (llist ∗supports, set ∗updated_combinations)
- void remove_changed_supports (set ∗updated_combinations)
- set ∗ new_updated_combinations (int combinations)
- void clear_support_tuple_sets (set ∗updated_combinations)
- void enum_table_pairs (htable ∗hta0, llist_node ∗combnp, int ∗vmap, comb_data ∗cd, comb_data ∗base_cd)
- int stage4_filter_with_no_check_insol_tuples (set ∗queue, int time, set ∗undo_set, set ∗cluster_vars, light_stack ∗undo_stack)
- int find_support_for_single_solution_bc (int t, comb_data ∗cd)
- int find_support_for_single_solution_bc_rec (int ∗∗tuples, comb_data ∗cd, int level, int ∗tuple_positions, int ∗tuple_position_htid)
- int find_support_for_single_solution_with_fc (int t, comb_data ∗cd, int first_table, int all_solutions, llist ∗remaining_vars, set ∗cluster_vars, active_tuple ∗ac, double ∗sol_count)
- int mark_all_in_future (int ∗old_marks, int mark_with, int var, llist ∗remaining_vars, int ∗current_tuple, comb_data ∗cd, llist ∗undo_dom, int ∗marks, llist ∗∗∗doms, int rec_depth, int drop_for_all_revised)
- int find_support_for_single_solution_with_fc_rec_single_out (int rec_depth, int ∗∗allocated_marks, int ∗old_marks, int ∗∗tuples, comb_data ∗cd, int var, int ∗tuple_positions, int ∗tuple_position_htid, llist ∗remaining_vars, int all_solutions, llist ∗∗∗doms, set ∗cluster_vars, active_tuple ∗ac, int priority_last_n, double ∗sol_count, llist ∗stack)
- int find_support_for_single_solution_with_fc_rec (int rec_depth, int ∗∗allocated_marks, int ∗old_marks, int ∗∗tuples, comb_data ∗cd, int var, int ∗tuple_positions, int ∗tuple_position_htid, llist ∗remaining_vars, int all_solutions, llist ∗∗∗doms, set ∗cluster_vars, active_tuple ∗ac, int priority_last_n, double ∗sol_count, llist ∗stack)
- int stage4_filter_with_all_solutions (set ∗queue, int time, set ∗undo_set, set ∗cluster_vars, light_stack ∗undo_stack, int ∗queues_total_count, int ppr)
- void find_all_supports_with_all_solutions_with_fc (comb_data ∗cd, set ∗cluster_varst, active_tuple ∗ac, double ∗sol_count)
- void set_dom (comb_data ∗cd)
- int choose_var (comb_data ∗cd, llist ∗remaining_vars, llist ∗undo_deg)
- int choose_var_priority (comb_data ∗cd, llist ∗remaining_vars, llist ∗undo_deg, int priroty_first_n)
- int choose_var_priority_decided (comb_data ∗cd, llist ∗remaining_vars, llist ∗undo_deg, int priority_last_n, int decided, active_tuple ∗ac)
- void undo_dom_deg_var (llist ∗undo_deg, llist ∗undo_dom, comb_data ∗cd, int var, llist ∗remaining_vars, llist ∗∗doms, llist ∗∗doms_xlevel)
- void set_deg_count (comb_data ∗cd, llist ∗remaining_vars, llist ∗undo_deg, int removed)
- htable ∗ get_ht_from_cd (comb_data ∗cd, int ht_pos)
- void set_revision_time (htable ∗ht, int t, int revision_time, comb_data ∗cd)
- void set_revision_time_parents (htable ∗ht, int t, int revision_time, comb_data ∗cd, llist ∗stack, int ∗∗tuples)
- int get_revision_time (htable ∗ht, int t)
- int visit_rest_of_tuples (llist ∗remaining_vars, int all_solutions, int next_var, comb_data ∗cd, int var, int ∗old_marks, int ∗∗tuples, llist ∗∗∗doms, int rec_depth, int ∗tuple_positions, llist ∗undo_deg, llist ∗undo_dom)
- int filter_all_relations_with_all_solutions (comb_data ∗cd, set ∗cluster_vars, active_tuple ∗ac, double ∗sol_count)
- int filter_single_relation_in_combination (comb_data ∗cd, set ∗undo_set, light_stack ∗undo_stack, double ∗sol_count)
- comb_data ∗ setup_relation_group_for_a_comb_first (llist ∗comb, int context, llist_node ∗firsttable, options category, comb_data ∗base_cd)
- comb_data ∗ setup_relation_group_for_a_comb (llist ∗comb, int context, options category, comb_data ∗base_cd)
- comb_data ∗ setup_relation_group_for_a_comb_long_indexes (llist ∗comb, options category)
- int filter_all_relations_with_all_solutions_priority (comb_data ∗cd, set ∗cluster_vars, active_tuple ∗ac, int priority_last_n, double ∗sol_count)
- int find_support_for_single_solution_with_fc_priority (int t, comb_data ∗cd, int first_table, int all_solutions, llist ∗remaining_vars, set ∗cluster_vars, active_tuple ∗ac, int priority_last_n, double ∗sol_count)

- void find_all_supports_with_all_solutions_with_fc_priority (comb_data ∗cd, set ∗cluster_vars, active_tuple ∗ac, int priority_last_n, double ∗sol_count)
- void find_all_supports_with_all_solutions_with_fc_priority_next_var (comb_data ∗cd, set ∗cluster_vars, active_tuple ∗ac, int priority_last_n, int next_v, double ∗sol_count)
- void queue_relation_related (htable ∗ht, set ∗queue)
- void queue_relation_related_multiqueue (htable ∗ht, set ∗queue, set ∗∗queues, int ∗queues_total_count)
- comb_cdata ∗ new_comb_cdata (int context)
- void destroy_comb_cdata (comb_cdata ∗ccd)
- llist ∗ get_comb_from (htable ∗filter, htable ∗given)
- void destroy_comb (llist ∗comb)
- void get_rstat (comb_data ∗cdata)
- void clear_comb_datas_from_hts (llist ∗htlist)

4.57.1 Detailed Description

Provides helper methods for initializing the data-structures for the PerTuple and AllSol algorithms.

4.57.2 Typedef Documentation

4.57.2.1 typedef struct **comb_cdata comb_cdata**

4.57.2.2 typedef struct **table_map table_map**

4.57.3 Function Documentation

4.57.3.1 void build_ind_trees (comb_data ∗ *cd,* **llist_node** ∗ *combnp,* int ∗ *vmap,* comb_data ∗ *base_cd*)

4.57.3.2 void build_joining_order (**htable** ∗ *hta,* **llist_node** ∗ *combnp,* int ∗ *vmap,* **llist** ∗ *hts_ordered,* **llist** ∗ *all_com_cols*)

4.57.3.3 int choose_var (comb_data ∗ *cd,* **llist** ∗ *remaining_vars,* **llist** ∗ *undo_deg*)

4.57.3.4 int choose_var_priority (comb_data ∗ *cd,* **llist** ∗ *remaining_vars,* **llist** ∗ *undo_deg,* int *priroty_first_n*)

4.57.3.5 int choose_var_priority_decided (comb_data ∗ *cd,* **llist** ∗ *remaining_vars,* **llist** ∗ *undo_deg,* int *priority_last_n,* int *decided,* **active_tuple** ∗ *ac*)

4.57.3.6 void clear_comb_datas_from_hts (**llist** ∗ *htlist*)

4.57.3.7 void clear_support_tuple_sets (**set** ∗ *updated_combinations*)

4.57.3.8 void destroy_comb (**llist** ∗ *comb*)

4.57.3.9 void destroy_comb_cdata (**comb_cdata** ∗ *ccd*)

4.57.3.10 void destroy_comb_data2 (comb_data ∗ *cd*)

4.57.3.11 void destroy_support (support ∗ *s*)

4.57.3.12 void enum_table_pairs (**htable** ∗ *hta0,* **llist_node** ∗ *combnp,* int ∗ *vmap,* comb_data ∗ *cd,* comb_data ∗ *base_cd*)

4.57.3.13 int filter_all_relations_with_all_solutions (comb_data ∗ *cd,* **set** ∗ *cluster_vars,* **active_tuple** ∗ *ac,* double ∗ *sol_count*)

4.57.3.14 int filter_all_relations_with_all_solutions_priority (comb_data * *cd,* set * *cluster_vars,* active_tuple * *ac,* int *priority_last_n,* double * *sol_count*)

4.57.3.15 void filter_relations_rs (llist * *queue*)

4.57.3.16 int filter_single_relation_in_combination (comb_data * *cd,* set * *undo_set,* light_stack * *undo_stack,* double * *sol_count*)

4.57.3.17 void find_all_supports_with_all_solutions_with_fc (comb_data * *cd,* set * *cluster_varst,* active_tuple * *ac,* double * *sol_count*)

4.57.3.18 void find_all_supports_with_all_solutions_with_fc_priority (comb_data * *cd,* set * *cluster_vars,* active_tuple * *ac,* int *priority_last_n,* double * *sol_count*)

4.57.3.19 void find_all_supports_with_all_solutions_with_fc_priority_next_var (comb_data * *cd,* set * *cluster_vars,* active_tuple * *ac,* int *priority_last_n,* int *next_v,* double * *sol_count*)

4.57.3.20 int find_support_for_single_solution (int *t,* comb_data * *cd,* set * *cluster_vars,* double * *sol_count*)

4.57.3.21 int find_support_for_single_solution_bc (int *t,* comb_data * *cd*)

4.57.3.22 int find_support_for_single_solution_bc_rec (int ** *tuples,* comb_data * *cd,* int *level,* int * *tuple_positions,* int * *tuple_position_htid*)

4.57.3.23 int find_support_for_single_solution_rec (int ** *tuples,* comb_data * *cd,* int *level,* int * *tuple_positions,* int * *tuple_position_htid*)

4.57.3.24 int find_support_for_single_solution_with_fc (int *t,* comb_data * *cd,* int *first_table,* int *all_solutions,* llist * *remaining_vars,* set * *cluster_vars,* active_tuple * *ac,* double * *sol_count*)

4.57.3.25 int find_support_for_single_solution_with_fc_priority (int *t,* comb_data * *cd,* int *first_table,* int *all_solutions,* llist * *remaining_vars,* set * *cluster_vars,* active_tuple * *ac,* int *priority_last_n,* double * *sol_count*)

4.57.3.26 int find_support_for_single_solution_with_fc_rec (int *rec_depth,* int ** *allocated_marks,* int * *old_marks,* int ** *tuples,* comb_data * *cd,* int *var,* int * *tuple_positions,* int * *tuple_position_htid,* llist * *remaining_vars,* int *all_solutions,* llist *** *doms,* set * *cluster_vars,* active_tuple * *ac,* int *priority_last_n,* double * *sol_count,* llist * *stack*)

4.57.3.27 int find_support_for_single_solution_with_fc_rec_single_out (int *rec_depth,* int ** *allocated_marks,* int * *old_marks,* int ** *tuples,* comb_data * *cd,* int *var,* int * *tuple_positions,* int * *tuple_position_htid,* llist * *remaining_vars,* int *all_solutions,* llist *** *doms,* set * *cluster_vars,* active_tuple * *ac,* int *priority_last_n,* double * *sol_count,* llist * *stack*)

4.57.3.28 llist* get_comb_from (htable * *filter,* htable * *given*)

4.57.3.29 htable* get_ht_from_cd (comb_data * *cd,* int *ht_pos*)

4.57.3.30 htable* get_hts_paired (comb_data * *cd,* int *hi,* int *hj,* int *p*)

4.57.3.31 llist* get_pair_common_cols (comb_data * *cd,* int *hi,* int *hj*)

4.57.3.32 int get_revision_time (htable * *ht,* int *t*)

4.57.3.33 void get_rstat (comb_data * *cdata*)

4.57.3.34 tree_map* get_tms_paired (comb_data * *cd,* int *hi,* int *hj*)

4.57.3.35 comb_data* make_comb_data2 (llist * comb, llist_node * combnp, int context, int no_tuple_positions, options category, comb_data * base_cd)

4.57.3.36 int mark_all_in_future (int * old_marks, int mark_with, int var, llist * remaining_vars, int * current_tuple, comb_data * cd, llist * undo_dom, int * marks, llist *** doms, int rec_depth, int drop_for_all_revised)

4.57.3.37 comb_cdata* new_comb_cdata (int context)

4.57.3.38 support* new_support (comb_data * comb, int tuple)

4.57.3.39 update_support* new_update_support (comb_data * comb)

4.57.3.40 set* new_updated_combinations (int combinations)

4.57.3.41 void queue_related_relations (set * queue, htable * ht)

4.57.3.42 void queue_relation_related (htable * ht, set * queue)

4.57.3.43 void queue_relation_related_multiqueue (htable * ht, set * queue, set ** queues, int * queues_total_count)

4.57.3.44 void queue_supported_tuples (llist * supports, set * updated_combinations)

4.57.3.45 void remove_changed_supports (set * updated_combinations)

4.57.3.46 void set_deg_count (comb_data * cd, llist * remaining_vars, llist * undo_deg, int removed)

4.57.3.47 void set_dom (comb_data * cd)

4.57.3.48 void set_revision_time (htable * ht, int t, int revision_time, comb_data * cd)

4.57.3.49 void set_revision_time_parents (htable * ht, int t, int revision_time, comb_data * cd, llist * stack, int ** tuples)

4.57.3.50 comb_data* setup_relation_group_for_a_comb (llist * comb, int context, options category, comb_data * base_cd)

4.57.3.51 comb_data* setup_relation_group_for_a_comb_first (llist * comb, int context, llist_node * firsttable, options category, comb_data * base_cd)

4.57.3.52 comb_data* setup_relation_group_for_a_comb_long_indexes (llist * comb, options category)

4.57.3.53 int setup_relation_groups (llist * combs, int context, options category)

4.57.3.54 int setup_relation_groups_ (llist * combs, int context, options category, comb_data ** cdatas, llist * context_list)

4.57.3.55 int stage4_filter (set * queue, int time, set * undo_set, set * cluster_vars, light_stack * undo_stack, int ppr)

4.57.3.56 int stage4_filter_auto (set * queue, set ** queues, int time, set * undo_set, set * cluster_vars, light_stack * undo_stack, int * queues_total_count, int ppr)

4.57.3.57 int stage4_filter_auto_all_sol (set * queue, set ** queues, int time, set * undo_set, light_stack * undo_stack, int * queues_total_count)

4.57.3.58 int stage4_filter_auto_one_sol (set * queue, set ** queues, int time, set * undo_set, light_stack * undo_stack, int * queues_total_count)

4.57.3.59 int stage4_filter_multiqueue (**set** ∗ *queue,* int *time,* **set** ∗ *undo_set,* **set** ∗ *cluster_vars,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int ∗ *queues_total_count,* int *ppr*)

4.57.3.60 int stage4_filter_with_all_solutions (**set** ∗ *queue,* int *time,* **set** ∗ *undo_set,* **set** ∗ *cluster_vars,* **light_stack** ∗ *undo_stack,* int ∗ *queues_total_count,* int *ppr*)

4.57.3.61 int stage4_filter_with_no_check_insol_tuples (**set** ∗ *queue,* int *time,* **set** ∗ *undo_set,* **set** ∗ *cluster_vars,* **light_stack** ∗ *undo_stack*)

4.57.3.62 **set**∗ stage4_init_queue (**llist** ∗ *constraint_graph_node_list*)

4.57.3.63 void undo_dom_deg_var (**llist** ∗ *undo_deg,* **llist** ∗ *undo_dom,* comb_data ∗ *cd,* int *var,* **llist** ∗ *remaining_vars,* **llist** ∗∗ *doms,* **llist** ∗∗ *doms_xlevel*)

4.57.3.64 int visit_rest_of_tuples (**llist** ∗ *remaining_vars,* int *all_solutions,* int *next_var,* comb_data ∗ *cd,* int *var,* int ∗ *old_marks,* int ∗∗ *tuples,* **llist** ∗∗∗ *doms,* int *rec_depth,* int ∗ *tuple_positions,* **llist** ∗ *undo_deg,* **llist** ∗ *undo_dom*)

4.58 rs2tree.c File Reference

```
#include "rs2tree.h"
#include <math.h>
#include "combinations_wrapper.h"
#include "rs2.h"
#include "rsprocedures.h"
#include "tree.h"
#include "btdallsol.h"
#include "domains.h"
#include "hashtable_search.h"
#include "decomposition_functions.h"
#include "decomposition.h"
#include "rr.h"
#include "tables.h"
#include "gbtd.h"
#include "septables.h"
#include "sort.h"
```

Functions

- int rs2_all_sols_bt_pre_processing (decomposed_tree ∗treen, rs2tree_configuration ∗config, int ppr)
- void pre_feature_extract (rb_tree ∗∗stats, decomposed_tree ∗treen)
- void post_feature_extract (rb_tree ∗stats, decomposed_tree ∗treen, int number_of_clusters)
- void rs2_all_sols_bt (decomposed_tree ∗treen)
- int process_tree_combinations (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗queue, int ppr)
- int process_tree_combinations2 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, int ∗repeat, int isroot, set ∗∗queues, int queues_count, int onelevel, int ppr)
- int process_tree_combinations3 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, int ∗repeat, int isroot, set ∗∗queues, int queues_count, int onelevel)
- rs2tree_stats ∗ get_make_stats (rb_tree ∗∗stats, decomposed_tree ∗treen)
- void init_update_processed_info (rb_tree ∗∗stats, decomposed_tree ∗treen, int init, int update)
- void init_processed_info (rb_tree ∗∗stats, decomposed_tree ∗treen)
- void update_processed_info (rb_tree ∗∗stats, decomposed_tree ∗treen)
- void rs2t_set_heap_pos (int val, void ∗body)
- void rs2t_set_heap_value (int val, void ∗body)

- int rs2t_get_heap_value (void ∗body)
- int rs2t_get_id (void ∗body)
- void check_this (set ∗∗queues, int count, int sum)
- decomposed_tree ∗ fringe_extract_top (heap ∗h)
- int process_tree_combinations5 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations52 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, light_stack ∗value_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations522 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, light_stack ∗value_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_this_cluster (decomposed_tree ∗current, set ∗undo_set, light_stack ∗undo_stack, light_stack ∗value_stack, set ∗∗queues, decomposed_tree ∗root, int ∗aqueues_total_count, int ppr)
- int process_tree_combinations53 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations54 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations6 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations7 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations8 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, light_stack ∗value_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations9 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations4 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, light_stack ∗value_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations4_ (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations4_filter_up_rec (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, int ppr)
- int process_tree_combinations4_filter_down_rec (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, int ppr)
- int process_tree_combinations4_filter (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, int ppr)
- void undo_tuple_deletions_for_rs2tree (set ∗undo_set, light_stack ∗undo_stack)
- void assign_table_context (llist ∗usedhts, decomposed_tree ∗treen)
- decomposed_tree ∗ find_top_root_with_scope (int ∗vars, decomposed_tree ∗root)
- int startBTD (decomposed_tree ∗treen, long double ∗total_sols_ret, set ∗queue, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, int print_sol)
- variable ∗ un_assign_variable (light_stack ∗assigned_var_stack, light_stack ∗undo_set_stack, light_stack ∗undo_tuple_stack, light_stack ∗undo_value_stack)
- void undo_value_deletions (light_stack ∗value_stack)
- int assign_variable (variable ∗var)
- variable ∗ choose_var_from_treen (decomposed_tree ∗treen)
- set ∗ set_variable_htables2 (decomposed_tree ∗∗c_array, int clusters)
- void set_queues_treenodes2_singleq (decomposed_tree ∗∗treens, int clusters, llist ∗tables, set ∗∗queues)
- void set_queues_treenodes2 (decomposed_tree ∗∗treens, int clusters, llist ∗tables, set ∗∗queues)
- void set_single_graph_combinations_to_treenodes (decomposed_tree ∗treen, llist ∗combs, constraint_graph ∗cg)
- void single_graph_combinations_with_tables (llist ∗tables, llist ∗∗ret_combs, constraint_graph ∗∗ret_cg)
- int filter_table_with_choosen_value (htable ∗ht, light_stack ∗undo_stack, set ∗undo_set, variable ∗var)
- void reset_variable_domain (variable ∗var)
- void reset_variable_domain2 (variable ∗var)
- void record_good_nogood (int good, long double weight, decomposed_tree ∗child)
- void set_good_nogood_trees (decomposed_tree ∗treen)
- void rs2tree_filter_domains (variables ∗vars, light_stack ∗value_stack)

- void reset_relation_queues (set ∗∗queues, int count)
- void set_cluster_constraints_to_projected_reols_on_c_vars (decomposed_tree ∗treen)
- void set_default_context_to_tables (decomposed_tree ∗treen)
- void ∗ get_payload (int param)
- void traverse_treemap_set_collect_counts (tree_map ∗tm, int width, llist ∗result, double norm_factor)
- void report_tree (decomposed_tree ∗treen, rb_tree ∗∗stats, int initial)
- double ∗ avgStdev2 (rs2tree_stats ∗st, double ∗∗array)
- double ∗ avgStdev (decomposed_tree ∗treen, double ∗∗array)
- rs2tree_stats ∗ new_rs2tree_stats (void)
- void extract_feature (rs2tree_stats ∗rs_stat, int ppr)
- void add_times_to_stats (rb_tree ∗stats)
- void print_rs2tree_stats (rb_tree ∗stats, int count, int thiscluster)
- int start_w_b_BTD (decomposed_tree ∗treen, long double ∗total_sols_ret, set ∗queue, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, int all_solutions)
- int rs2tree_filter_relations_given_domains (llist ∗separatorvars, light_stack ∗undo_stack, light_stack ∗value_stack, set ∗undo_set, set ∗queue, set ∗∗queues, int ∗queues_total_count, llist ∗propagator_vars)
- void rs2tree_filter_domains_given_relations (llist ∗constraints, llist ∗separatorvars, light_stack ∗value_stack)
- void rs2tree_filter_domains_given_relation (htable ∗rel, light_stack ∗value_stack, llist ∗propagator_vars)
- int process_geneate_bolstering_bin_cliqueue_relations (decomposed_tree ∗root)

4.58.1 Function Documentation

4.58.1.1 void add_times_to_stats (**rb_tree** ∗ *stats*)

4.58.1.2 void assign_table_context (**llist** ∗ *usedhts,* **decomposed_tree** ∗ *treen*)

4.58.1.3 int assign_variable (**variable** ∗ *var*)

4.58.1.4 double∗ avgStdev (**decomposed_tree** ∗ *treen,* double ∗∗ *array*)

4.58.1.5 double∗ avgStdev2 (**rs2tree_stats** ∗ *st,* double ∗∗ *array*)

4.58.1.6 void check_this (**set** ∗∗ *queues,* int *count,* int *sum*)

4.58.1.7 **variable**∗ choose_var_from_treen (**decomposed_tree** ∗ *treen*)

4.58.1.8 void extract_feature (**rs2tree_stats** ∗ *rs_stat,* int *ppr*)

4.58.1.9 int filter_table_with_choosen_value (**htable** ∗ *ht,* **light_stack** ∗ *undo_stack,* **set** ∗ *undo_set,* **variable** ∗ *var*)

4.58.1.10 **decomposed_tree** ∗ find_top_root_with_scope (int ∗ *vars,* **decomposed_tree** ∗ *root*)

4.58.1.11 **decomposed_tree**∗ fringe_extract_top (**heap** ∗ *h*)

4.58.1.12 **rs2tree_stats**∗ get_make_stats (**rb_tree** ∗∗ *stats,* **decomposed_tree** ∗ *treen*)

4.58.1.13 void∗ get_payload (int *param*)

4.58.1.14 void init_processed_info (**rb_tree** ∗∗ *stats,* **decomposed_tree** ∗ *treen*)

4.58.1.15 void init_update_processed_info (**rb_tree** ∗∗ *stats,* **decomposed_tree** ∗ *treen,* int *init,* int *update*)

4.58.1.16 **rs2tree_stats**∗ new_rs2tree_stats (void)

4.58.1.17 void post_feature_extract (**rb_tree** ∗ *stats,* **decomposed_tree** ∗ *treen,* int *number_of_clusters*)

4.58.1.18 void pre_feature_extract (**rb_tree** ∗∗ *stats,* **decomposed_tree** ∗ *treen*)

4.58.1.19 void print_rs2tree_stats (**rb_tree** ∗ *stats,* int *count,* int *thiscluster*)

4.58.1.20 int process_geneate_bolstering_bin_cliqueue_relations (**decomposed_tree** ∗ *root*)

4.58.1.21 int process_this_cluster (**decomposed_tree** ∗ *current,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **light_stack** ∗ *value_stack,* **set** ∗∗ *queues,* **decomposed_tree** ∗ *root,* int ∗ *aqueues_total_count,* int *ppr*)

4.58.1.22 int process_tree_combinations (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗ *queue,* int *ppr*)

4.58.1.23 int process_tree_combinations2 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* int ∗ *repeat,* int *isroot,* **set** ∗∗ *queues,* int *queues_count,* int *onelevel,* int *ppr*)

4.58.1.24 int process_tree_combinations3 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* int ∗ *repeat,* int *isroot,* **set** ∗∗ *queues,* int *queues_count,* int *onelevel*)

4.58.1.25 int process_tree_combinations4 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **light_stack** ∗ *value_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.58.1.26 int process_tree_combinations4_ (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.58.1.27 int process_tree_combinations4_filter (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int *queues_count,* int *ppr*)

4.58.1.28 int process_tree_combinations4_filter_down_rec (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int *queues_count,* int *ppr*)

4.58.1.29 int process_tree_combinations4_filter_up_rec (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int *queues_count,* int *ppr*)

4.58.1.30 int process_tree_combinations5 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.58.1.31 int process_tree_combinations52 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **light_stack** ∗ *value_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.58.1.32 int process_tree_combinations522 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **light_stack** ∗ *value_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.58.1.33 int process_tree_combinations53 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.58.1.34 int process_tree_combinations54 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.58.1.35 int process_tree_combinations6 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.58.1.36 int process_tree_combinations7 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.58.1.37 int process_tree_combinations8 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **light_stack** ∗ *value_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.58.1.38 int process_tree_combinations9 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.58.1.39 void record_good_nogood (int *good,* long double *weight,* **decomposed_tree** ∗ *child*)

4.58.1.40 void report_tree (**decomposed_tree** ∗ *treen,* **rb_tree** ∗∗ *stats,* int *initial*)

for (np = treen->children->head; np != NULL; np = np->next) { decomposed_tree ∗ child = (decomposed_-tree∗)np->body; report_tree (child, stats, initial, queues); }

4.58.1.41 void reset_relation_queues (**set** ∗∗ *queues,* int *count*)

4.58.1.42 void reset_variable_domain (**variable** ∗ *var*)

4.58.1.43 void reset_variable_domain2 (**variable** ∗ *var*)

4.58.1.44 void rs2_all_sols_bt (**decomposed_tree** ∗ *treen*)

4.58.1.45 int rs2_all_sols_bt_pre_processing (**decomposed_tree** ∗ *treen,* **rs2tree_configuration** ∗ *config,* int *ppr*)

4.58.1.46 int rs2t_get_heap_value (void ∗ *body*)

4.58.1.47 int rs2t_get_id (void ∗ *body*)

4.58.1.48 void rs2t_set_heap_pos (int *val,* void ∗ *body*)

4.58.1.49 void rs2t_set_heap_value (int *val,* void ∗ *body*)

4.58.1.50 void rs2tree_filter_domains (**variables** ∗ *vars,* **light_stack** ∗ *value_stack*)

4.58.1.51 void rs2tree_filter_domains_given_relation (**htable** ∗ *rel,* **light_stack** ∗ *value_stack,* **llist** ∗ *propagator_vars*)

4.58.1.52 void rs2tree_filter_domains_given_relations (**llist** ∗ *constraints,* **llist** ∗ *separatorvars,* **light_stack** ∗ *value_stack*)

4.58.1.53 int rs2tree_filter_relations_given_domains (**llist** ∗ *separatorvars,* **light_stack** ∗ *undo_stack,* **light_stack** ∗ *value_stack,* **set** ∗ *undo_set,* **set** ∗ *queue,* **set** ∗∗ *queues,* int ∗ *queues_total_count,* **llist** ∗ *propagator_vars*)

4.58.1.54 void set_cluster_constraints_to_projected_reols_on_c_vars (**decomposed_tree** ∗ *treen*)

4.58.1.55 void set_default_context_to_tables (**decomposed_tree** ∗ *treen*)

4.58.1.56 void set_good_nogood_trees (**decomposed_tree** ∗ *treen*)

4.58.1.57 void set_queues_treenodes2 (**decomposed_tree** ∗∗ *treens,* int *clusters,* **llist** ∗ *tables,* **set** ∗∗ *queues*)

4.58.1.58 void set_queues_treenodes2_singleq (**decomposed_tree** ∗∗ *treens,* int *clusters,* **llist** ∗ *tables,* **set** ∗∗ *queues*)

4.58.1.59 void set_single_graph_combinations_to_treenodes (**decomposed_tree** ∗ *treen,* **llist** ∗ *combs,* **constraint_graph** ∗ *cg*)

4.58.1.60 **set**∗ set_variable_htables2 (**decomposed_tree** ∗∗ *c_array,* int *clusters*)

4.58.1.61 void single_graph_combinations_with_tables (**llist** ∗ *tables,* **llist** ∗∗ *ret_combs,* **constraint_graph** ∗∗ *ret_cg*)

4.58.1.62 int start_w_b_BTD (**decomposed_tree** ∗ *treen,* long double ∗ *total_sols_ret,* **set** ∗ *queue,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* int *all_solutions*)

4.58.1.63 int startBTD (**decomposed_tree** ∗ *treen,* long double ∗ *total_sols_ret,* **set** ∗ *queue,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* int *print_sol*)

4.58.1.64 void traverse_treemap_set_collect_counts (**tree_map** ∗ *tm,* int *width,* **llist** ∗ *result,* double *norm_factor*)

4.58.1.65 **variable**∗ un_assign_variable (**light_stack** ∗ *assigned_var_stack,* **light_stack** ∗ *undo_set_stack,* **light_stack** ∗ *undo_tuple_stack,* **light_stack** ∗ *undo_value_stack*)

4.58.1.66 void undo_tuple_deletions_for_rs2tree (**set** ∗ *undo_set,* **light_stack** ∗ *undo_stack*)

4.58.1.67 void undo_value_deletions (**light_stack** ∗ *value_stack*)

4.58.1.68 void update_processed_info (**rb_tree** ∗∗ *stats,* **decomposed_tree** ∗ *treen*)

4.59 rs2tree.h File Reference

```
#include "decomposition.h"
#include "llist.h"
#include "query_graph.h"
```

Classes

- struct rs2tree_configuration
- struct rs2tree_stats

Typedefs

- typedef struct rs2tree_configuration rs2tree_configuration
- typedef struct rs2tree_stats rs2tree_stats

Functions

- rs2tree_stats ∗ new_rs2tree_stats (void)
- void set_combinations (decomposed_tree ∗treen, set ∗∗queues)
- int set_combinations2 (decomposed_tree ∗treen, set ∗∗queues)
- int process_tree_combinations (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗queue, int ppr)
- int process_tree_combinations2 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, int ∗repeat, int isroot, set ∗∗queues, int queues_count, int onelevel, int ppr)
- void undo_tuple_deletions_for_rs2tree (set ∗undo_set, light_stack ∗undo_stack)
- int startBTD (decomposed_tree ∗treen, long double ∗total_sols_ret, set ∗queue, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, int print_sol)
- variable ∗ choose_var_from_treen (decomposed_tree ∗treen)
- variable ∗ un_assign_variable (light_stack ∗assigned_var_stack, light_stack ∗undo_set_stack, light_stack ∗undo_tuple_stack, light_stack ∗undo_value_stack)
- int assign_variable (variable ∗var)
- int filter_table_with_choosen_value (htable ∗ht, light_stack ∗undo_stack, set ∗undo_set, variable ∗var)

- void undo_value_deletions (light_stack ∗undo_value_stack)
- int rs2_all_sols_bt_pre_processing (decomposed_tree ∗treen, rs2tree_configuration ∗config, int ppr)
- void rs2_all_sols_bt (decomposed_tree ∗treen)
- void reset_variable_domain (variable ∗var)
- void reset_variable_domain2 (variable ∗var)
- void single_graph_combinations_with_tables (llist ∗tables, llist ∗∗ret_combs, constraint_graph ∗∗ret_cg)
- void set_single_graph_combinations_to_treenodes (decomposed_tree ∗treen, llist ∗combs, constraint_graph ∗cg)
- void record_good_nogood (int good, long double value, decomposed_tree ∗child)
- void set_good_nogood_trees (decomposed_tree ∗treen)
- void rs2tree_filter_domains (variables ∗vars, light_stack ∗value_stack)
- void reset_relation_queues (set ∗∗queues, int count)
- void set_cluster_constraints_to_projected_reols_on_c_vars (decomposed_tree ∗treen)
- decomposed_tree ∗ find_top_root_with_scope (int ∗vars, decomposed_tree ∗root)
- void assign_table_context (llist ∗usedhts, decomposed_tree ∗treen)
- int set_combinations3 (decomposed_tree ∗treen, set ∗∗queues)
- void set_queues_treenodes (decomposed_tree ∗treen, set ∗∗queues)
- void set_queues_treenodes2 (decomposed_tree ∗∗treens, int clusters, llist ∗tables, set ∗∗queues)
- void set_queues_treenodes2_singleq (decomposed_tree ∗∗treens, int clusters, llist ∗tables, set ∗∗queues)
- int process_this_cluster (decomposed_tree ∗current, set ∗undo_set, light_stack ∗undo_stack, light_stack ∗value_stack, set ∗∗queues, decomposed_tree ∗root, int ∗aqueues_total_count, int ppr)
- int process_tree_combinations3 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, int ∗repeat, int isroot, set ∗∗queues, int queues_count, int onelevel)
- int process_tree_combinations5 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations52 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, light_stack ∗value_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations522 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, light_stack ∗value_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations53 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations54 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations6 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations7 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations8 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, light_stack ∗value_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations9 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations4 (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, light_stack ∗value_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations4_ (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, variable ∗var, int ppr)
- int process_tree_combinations4_filter_up_rec (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, int ppr)
- int process_tree_combinations4_filter_down_rec (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, int ppr)
- int process_tree_combinations4_filter (decomposed_tree ∗treen, set ∗undo_set, light_stack ∗undo_stack, set ∗∗queues, int queues_count, int ppr)
- set ∗ collect_htables (decomposed_tree ∗treen, set ∗usedhts, int separator_rels_also)
- set ∗ set_variable_htables (decomposed_tree ∗treen, set ∗usedhts, int separator_rels_also)
- set ∗ set_variable_htables2 (decomposed_tree ∗∗c_array, int clusters)
- void generate_combinations_per_cluster (decomposed_tree ∗treen)
- void set_default_context_to_tables (decomposed_tree ∗treen)

- void report_tree (decomposed_tree ∗treen, rb_tree ∗∗stats, int initial)
- void add_times_to_stats (rb_tree ∗stats)
- void print_rs2tree_stats (rb_tree ∗stats, int count, int thiscluster)
- int start_w_b_BTD (decomposed_tree ∗treen, long double ∗total_sols_ret, set ∗queue, set ∗∗queues, int queues_count, decomposed_tree ∗∗treens, int all_solutions)
- double ∗ avgStdev (decomposed_tree ∗treen, double ∗∗array)
- double ∗ avgStdev2 (rs2tree_stats ∗st, double ∗∗array)
- void ∗ get_payload (int param)
- void traverse_treemap_set_collect_counts (tree_map ∗tm, int width, llist ∗result, double norm_factor)
- rs2tree_stats ∗ get_make_stats (rb_tree ∗∗stats, decomposed_tree ∗treen)
- void init_processed_info (rb_tree ∗∗stats, decomposed_tree ∗treen)
- void update_processed_info (rb_tree ∗∗stats, decomposed_tree ∗treen)
- void init_update_processed_info (rb_tree ∗∗stats, decomposed_tree ∗treen, int init, int update)
- void rs2t_set_heap_pos (int val, void ∗body)
- void rs2t_set_heap_value (int val, void ∗body)
- int rs2t_get_heap_value (void ∗body)
- decomposed_tree ∗ fringe_extract_top (heap ∗h)
- int rs2t_get_id (void ∗body)
- void check_this (set ∗∗queues, int count, int sum)
- void extract_feature (rs2tree_stats ∗rs_stat, int ppr)
- int rs2tree_filter_relations_given_domains (llist ∗separatorvars, light_stack ∗undo_stack, light_stack ∗value_stack, set ∗undo_set, set ∗queue, set ∗∗queues, int ∗queues_total_count, llist ∗propagator_vars)
- void rs2tree_filter_domains_given_relations (llist ∗constraints, llist ∗separatorvars, light_stack ∗value_stack)
- void rs2tree_filter_domains_given_relation (htable ∗rel, light_stack ∗value_stack, llist ∗propagator_vars)
- int process_geneate_bolstering_bin_cliqueue_relations (decomposed_tree ∗root)
- void pre_feature_extract (rb_tree ∗∗stats, decomposed_tree ∗treen)
- void post_feature_extract (rb_tree ∗stats, decomposed_tree ∗treen, int number_of_clusters)

4.59.1 Detailed Description

Implements the interfaces for the structure-based PerTuple and AllSol algorithms.

4.59.2 Typedef Documentation

4.59.2.1 typedef struct **rs2tree_configuration rs2tree_configuration**

4.59.2.2 typedef struct **rs2tree_stats rs2tree_stats**

4.59.3 Function Documentation

4.59.3.1 void add_times_to_stats (**rb_tree** ∗ *stats*)

4.59.3.2 void assign_table_context (**llist** ∗ *usedhts,* **decomposed_tree** ∗ *treen*)

4.59.3.3 int assign_variable (**variable** ∗ *var*)

4.59.3.4 double∗ avgStdev (**decomposed_tree** ∗ *treen,* double ∗∗ *array*)

4.59.3.5 double∗ avgStdev2 (**rs2tree_stats** ∗ *st,* double ∗∗ *array*)

4.59.3.6 void check_this (**set** ∗∗ *queues,* int *count,* int *sum*)

4.59.3.7 **variable**∗ choose_var_from_treen (**decomposed_tree** ∗ *treen*)

4.59.3.8 set∗ collect_htables (decomposed_tree ∗ *treen,* set ∗ *usedhts,* int *separator_rels_also*)

4.59.3.9 void extract_feature (rs2tree_stats ∗ *rs_stat,* int *ppr*)

4.59.3.10 int filter_table_with_choosen_value (htable ∗ *ht,* light_stack ∗ *undo_stack,* set ∗ *undo_set,* variable ∗ *var*)

4.59.3.11 decomposed_tree∗ find_top_root_with_scope (int ∗ *vars,* decomposed_tree ∗ *root*)

4.59.3.12 decomposed_tree∗ fringe_extract_top (heap ∗ *h*)

4.59.3.13 void generate_combinations_per_cluster (decomposed_tree ∗ *treen*)

4.59.3.14 rs2tree_stats∗ get_make_stats (rb_tree ∗∗ *stats,* decomposed_tree ∗ *treen*)

4.59.3.15 void∗ get_payload (int *param*)

4.59.3.16 void init_processed_info (rb_tree ∗∗ *stats,* decomposed_tree ∗ *treen*)

4.59.3.17 void init_update_processed_info (rb_tree ∗∗ *stats,* decomposed_tree ∗ *treen,* int *init,* int *update*)

4.59.3.18 rs2tree_stats∗ new_rs2tree_stats (void)

4.59.3.19 void post_feature_extract (rb_tree ∗ *stats,* decomposed_tree ∗ *treen,* int *number_of_clusters*)

4.59.3.20 void pre_feature_extract (rb_tree ∗∗ *stats,* decomposed_tree ∗ *treen*)

4.59.3.21 void print_rs2tree_stats (rb_tree ∗ *stats,* int *count,* int *thiscluster*)

4.59.3.22 int process_geneate_bolstering_bin_cliqueue_relations (decomposed_tree ∗ *root*)

4.59.3.23 int process_this_cluster (decomposed_tree ∗ *current,* set ∗ *undo_set,* light_stack ∗ *undo_stack,* light_stack ∗ *value_stack,* set ∗∗ *queues,* decomposed_tree ∗ *root,* int ∗ *aqueues_total_count,* int *ppr*)

4.59.3.24 int process_tree_combinations (decomposed_tree ∗ *treen,* set ∗ *undo_set,* light_stack ∗ *undo_stack,* set ∗ *queue,* int *ppr*)

4.59.3.25 int process_tree_combinations2 (decomposed_tree ∗ *treen,* set ∗ *undo_set,* light_stack ∗ *undo_stack,* int ∗ *repeat,* int *isroot,* set ∗∗ *queues,* int *queues_count,* int *onelevel,* int *ppr*)

4.59.3.26 int process_tree_combinations3 (decomposed_tree ∗ *treen,* set ∗ *undo_set,* light_stack ∗ *undo_stack,* int ∗ *repeat,* int *isroot,* set ∗∗ *queues,* int *queues_count,* int *onelevel*)

4.59.3.27 int process_tree_combinations4 (decomposed_tree ∗ *treen,* set ∗ *undo_set,* light_stack ∗ *undo_stack,* light_stack ∗ *value_stack,* set ∗∗ *queues,* int *queues_count,* decomposed_tree ∗∗ *treens,* variable ∗ *var,* int *ppr*)

4.59.3.28 int process_tree_combinations4_ (decomposed_tree ∗ *treen,* set ∗ *undo_set,* light_stack ∗ *undo_stack,* set ∗∗ *queues,* int *queues_count,* decomposed_tree ∗∗ *treens,* variable ∗ *var,* int *ppr*)

4.59.3.29 int process_tree_combinations4_filter (decomposed_tree ∗ *treen,* set ∗ *undo_set,* light_stack ∗ *undo_stack,* set ∗∗ *queues,* int *queues_count,* int *ppr*)

4.59.3.30 int process_tree_combinations4_filter_down_rec (decomposed_tree ∗ *treen,* set ∗ *undo_set,* light_stack ∗ *undo_stack,* set ∗∗ *queues,* int *queues_count,* int *ppr*)

4.59.3.31 int process_tree_combinations4_filter_up_rec (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int *queues_count,* int *ppr*)

4.59.3.32 int process_tree_combinations5 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.59.3.33 int process_tree_combinations52 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **light_stack** ∗ *value_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.59.3.34 int process_tree_combinations522 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **light_stack** ∗ *value_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.59.3.35 int process_tree_combinations53 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.59.3.36 int process_tree_combinations54 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.59.3.37 int process_tree_combinations6 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.59.3.38 int process_tree_combinations7 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.59.3.39 int process_tree_combinations8 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **light_stack** ∗ *value_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.59.3.40 int process_tree_combinations9 (**decomposed_tree** ∗ *treen,* **set** ∗ *undo_set,* **light_stack** ∗ *undo_stack,* **set** ∗∗ *queues,* int *queues_count,* **decomposed_tree** ∗∗ *treens,* **variable** ∗ *var,* int *ppr*)

4.59.3.41 void record_good_nogood (int *good,* long double *value,* **decomposed_tree** ∗ *child*)

4.59.3.42 void report_tree (**decomposed_tree** ∗ *treen,* **rb_tree** ∗∗ *stats,* int *initial*)

for (np = treen->children->head; np != NULL; np = np->next) { decomposed_tree ∗ child = (decomposed_tree∗)np->body; report_tree (child, stats, initial, queues); }

4.59.3.43 void reset_relation_queues (**set** ∗∗ *queues,* int *count*)

4.59.3.44 void reset_variable_domain (**variable** ∗ *var*)

4.59.3.45 void reset_variable_domain2 (**variable** ∗ *var*)

4.59.3.46 void rs2_all_sols_bt (**decomposed_tree** ∗ *treen*)

4.59.3.47 int rs2_all_sols_bt_pre_processing (**decomposed_tree** ∗ *treen,* **rs2tree_configuration** ∗ *config,* int *ppr*)

4.59.3.48 int rs2t_get_heap_value (void ∗ *body*)

4.59.3.49 int rs2t_get_id (void ∗ *body*)

4.59.3.50 void rs2t_set_heap_pos (int *val,* void ∗ *body*)

4.59.3.51 void rs2t_set_heap_value (int *val,* void ∗ *body*)

4.59.3.52 void rs2tree_filter_domains (variables ∗ *vars,* light_stack ∗ *value_stack*)

4.59.3.53 void rs2tree_filter_domains_given_relation (htable ∗ *rel,* light_stack ∗ *value_stack,* llist ∗ *propagator_vars*)

4.59.3.54 void rs2tree_filter_domains_given_relations (llist ∗ *constraints,* llist ∗ *separatorvars,* light_stack ∗ *value_stack*)

4.59.3.55 int rs2tree_filter_relations_given_domains (llist ∗ *separatorvars,* light_stack ∗ *undo_stack,* light_stack ∗ *value_stack,* set ∗ *undo_set,* set ∗ *queue,* set ∗∗ *queues,* int ∗ *queues_total_count,* llist ∗ *propagator_vars*)

4.59.3.56 void set_cluster_constraints_to_projected_reols_on_c_vars (decomposed_tree ∗ *treen*)

4.59.3.57 void set_combinations (decomposed_tree ∗ *treen,* set ∗∗ *queues*)

4.59.3.58 int set_combinations2 (decomposed_tree ∗ *treen,* set ∗∗ *queues*)

4.59.3.59 int set_combinations3 (decomposed_tree ∗ *treen,* set ∗∗ *queues*)

4.59.3.60 void set_default_context_to_tables (decomposed_tree ∗ *treen*)

4.59.3.61 void set_good_nogood_trees (decomposed_tree ∗ *treen*)

4.59.3.62 void set_queues_treenodes (decomposed_tree ∗ *treen,* set ∗∗ *queues*)

4.59.3.63 void set_queues_treenodes2 (decomposed_tree ∗∗ *treens,* int *clusters,* llist ∗ *tables,* set ∗∗ *queues*)

4.59.3.64 void set_queues_treenodes2_singleq (decomposed_tree ∗∗ *treens,* int *clusters,* llist ∗ *tables,* set ∗∗ *queues*)

4.59.3.65 void set_single_graph_combinations_to_treenodes (decomposed_tree ∗ *treen,* llist ∗ *combs,* constraint_graph ∗ *cg*)

4.59.3.66 set∗ set_variable_htables (decomposed_tree ∗ *treen,* set ∗ *usedhts,* int *separator_rels_also*)

4.59.3.67 set∗ set_variable_htables2 (decomposed_tree ∗∗ *c_array,* int *clusters*)

4.59.3.68 void single_graph_combinations_with_tables (llist ∗ *tables,* llist ∗∗ *ret_combs,* constraint_graph ∗∗ *ret_cg*)

4.59.3.69 int start_w_b_BTD (decomposed_tree ∗ *treen,* long double ∗ *total_sols_ret,* set ∗ *queue,* set ∗∗ *queues,* int *queues_count,* decomposed_tree ∗∗ *treens,* int *all_solutions*)

4.59.3.70 int startBTD (decomposed_tree ∗ *treen,* long double ∗ *total_sols_ret,* set ∗ *queue,* set ∗∗ *queues,* int *queues_count,* decomposed_tree ∗∗ *treens,* int *print_sol*)

4.59.3.71 void traverse_treemap_set_collect_counts (tree_map ∗ *tm,* int *width,* llist ∗ *result,* double *norm_factor*)

4.59.3.72 variable∗ un_assign_variable (light_stack ∗ *assigned_var_stack,* light_stack ∗ *undo_set_stack,* light_stack ∗ *undo_tuple_stack,* light_stack ∗ *undo_value_stack*)

4.59.3.73 void undo_tuple_deletions_for_rs2tree (set ∗ *undo_set,* light_stack ∗ *undo_stack*)

4.59.3.74 void undo_value_deletions (light_stack ∗ *undo_value_stack*)

4.59.3.75 void update_processed_info (rb_tree ∗∗ *stats,* decomposed_tree ∗ *treen*)

4.60 septables.c File Reference

```
#include "septables.h"
#include "utils.h"
#include "partition.h"
#include "tables.h"
#include "hashtable_search.h"
#include "active_tuple.h"
#include "rb_tree.h"
```

Functions

- int add_maxclique_scopes (decomposed_tree ∗root)
- int normalize_generate_separator_scopes_relations (decomposed_tree ∗root)
- int add_binary_scopes (decomposed_tree ∗root)
- int shape_constraint_scopes_pE (decomposed_tree ∗clique)
- int normalize_small_big_lists (llist ∗bigger_new_constraints, llist ∗smaller_old_constraints, decomposed_tree ∗clique)
- void shape_constraint_scopes_pE_global (decomposed_tree ∗clique)

Variables

- int varmap_s = 0

4.60.1 Function Documentation

4.60.1.1 int add_binary_scopes (**decomposed_tree** ∗ *root*)

4.60.1.2 int add_maxclique_scopes (**decomposed_tree** ∗ *root*)

4.60.1.3 int normalize_generate_separator_scopes_relations (**decomposed_tree** ∗ *root*)

4.60.1.4 int normalize_small_big_lists (**llist** ∗ *bigger_new_constraints,* **llist** ∗ *smaller_old_constraints,* **decomposed_tree** ∗ *clique*)

4.60.1.5 int shape_constraint_scopes_pE (**decomposed_tree** ∗ *clique*)

4.60.1.6 void shape_constraint_scopes_pE_global (**decomposed_tree** ∗ *clique*)

4.60.2 Variable Documentation

4.60.2.1 int varmap_s = 0

4.61 septables.h File Reference

```
#include "decomposition.h"
```

Functions

- int add_maxclique_scopes (decomposed_tree ∗root)
- int add_binary_scopes (decomposed_tree ∗root)

- int shape_constraint_scopes_pE (decomposed_tree *clique)
- void shape_constraint_scopes_pE_global (decomposed_tree *clique)
- int normalize_generate_separator_scopes_relations (decomposed_tree *root)
- int normalize_small_big_lists (llist *bigger_new_constraints, llist *smaller_old_constraints, decomposed_tree *clique)

4.61.1 Detailed Description

Implements helper methods for generating redundant constraints at the separators.

4.61.2 Function Documentation

4.61.2.1 int add_binary_scopes (**decomposed_tree** ∗ *root*)

4.61.2.2 int add_maxclique_scopes (**decomposed_tree** ∗ *root*)

4.61.2.3 int normalize_generate_separator_scopes_relations (**decomposed_tree** ∗ *root*)

4.61.2.4 int normalize_small_big_lists (**llist** ∗ *bigger_new_constraints,* **llist** ∗ *smaller_old_constraints,* **decomposed_tree** ∗ *clique*)

4.61.2.5 int shape_constraint_scopes_pE (**decomposed_tree** ∗ *clique*)

4.61.2.6 void shape_constraint_scopes_pE_global (**decomposed_tree** ∗ *clique*)

4.62 set.c File Reference

```
#include "set.h"
#include "utils.h"
#include "rb_tree.h"
```

Functions

- s_node ∗ new_s_node (int key, void ∗body)
- set ∗ copy_set (set ∗s)
- set ∗ new_set (int size)
- set ∗ new_set_nomap (int size)
- void reset_set (set ∗s)
- set ∗ new_set_map (int size)
- llist ∗ set_to_list (set ∗s)
- void destroy_set (set ∗s)
- int add_int_element (set ∗s, int key)
- int get_int_element (set ∗s, int key)
- int remove_int_element (set ∗s, int key)
- int add_element (set ∗s, void ∗e, int key)
- int add_element_nomap (set ∗s, void ∗e, int key)
- void add_element_node (set ∗s, s_node ∗n)
- void ∗ remove_element (set ∗s)
- void ∗ peek_element (set ∗s)
- void ∗ remove_element_with_key (set ∗s, int key, int get_key(void ∗))
- void ∗ get_element_with_key (set ∗s, int key, int get_key(void ∗))
- int remove_element_key (set ∗s)
- int set_empty (set ∗s)

4.62.1 Function Documentation

4.62.1.1 int add_element (**set** ∗ *s,* void ∗ *e,* int *key*)

4.62.1.2 void add_element_node (**set** ∗ *s,* **s_node** ∗ *n*)

4.62.1.3 int add_element_nomap (**set** ∗ *s,* void ∗ *e,* int *key*)

4.62.1.4 int add_int_element (**set** ∗ *s,* int *key*)

4.62.1.5 **set**∗ copy_set (**set** ∗ *s*)

4.62.1.6 void destroy_set (**set** ∗ *s*)

4.62.1.7 void∗ get_element_with_key (**set** ∗ *s,* int *key,* int *get_keyvoid* ∗)

4.62.1.8 int get_int_element (**set** ∗ *s,* int *key*)

4.62.1.9 **s_node**∗ new_s_node (int *key,* void ∗ *body*)

4.62.1.10 **set**∗ new_set (int *size*)

4.62.1.11 **set**∗ new_set_map (int *size*)

4.62.1.12 **set**∗ new_set_nomap (int *size*)

4.62.1.13 void∗ peek_element (**set** ∗ *s*)

4.62.1.14 void∗ remove_element (**set** ∗ *s*)

4.62.1.15 int remove_element_key (**set** ∗ *s*)

4.62.1.16 void∗ remove_element_with_key (**set** ∗ *s,* int *key,* int *get_keyvoid* ∗)

4.62.1.17 int remove_int_element (**set** ∗ *s,* int *key*)

4.62.1.18 void reset_set (**set** ∗ *s*)

4.62.1.19 int set_empty (**set** ∗ *s*)

4.62.1.20 **llist**∗ set_to_list (**set** ∗ *s*)

4.63 set.h File Reference

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

Classes

- struct s_node
- struct set

Typedefs

- typedef struct s_node s_node
- typedef struct set set

Functions

- s_node ∗ new_s_node (int key, void ∗body)
- set ∗ new_set (int size)
- set ∗ new_set_map (int size)
- set ∗ new_set_nomap (int size)
- int add_element (set ∗s, void ∗e, int key)
- int add_element_nomap (set ∗s, void ∗e, int key)
- void add_element_node (set ∗s, s_node ∗n)
- void ∗ remove_element (set ∗s)
- int remove_element_key (set ∗s)
- int set_empty (set ∗s)
- void destroy_set (set ∗s)
- void ∗ peek_element (set ∗s)
- void ∗ remove_element_with_key (set ∗s, int key, int get_key(void ∗))
- void ∗ get_element_with_key (set ∗s, int key, int get_key(void ∗))
- int remove_int_element (set ∗s, int key)
- int get_int_element (set ∗s, int key)
- int add_int_element (set ∗s, int key)
- set ∗ copy_set (set ∗s)
- void reset_set (set ∗s)
- llist ∗ set_to_list (set ∗s)

4.63.1 Detailed Description

Provides a naive set implementation.

4.63.2 Typedef Documentation

4.63.2.1 typedef struct **s_node s_node**

4.63.2.2 typedef struct **set set**

4.63.3 Function Documentation

4.63.3.1 int add_element (**set** ∗ ***s,*** void ∗ ***e,*** int ***key***)

4.63.3.2 void add_element_node (**set** ∗ ***s,*** **s_node** ∗ ***n***)

4.63.3.3 int add_element_nomap (**set** ∗ ***s,*** void ∗ ***e,*** int ***key***)

4.63.3.4 int add_int_element (**set** ∗ ***s,*** int ***key***)

4.63.3.5 **set**∗ copy_set (**set** ∗ ***s***)

4.63.3.6 void destroy_set (**set** ∗ ***s***)

4.63.3.7 void∗ get_element_with_key (**set** ∗ ***s,*** int ***key,*** int ***get_keyvoid*** ∗)

4.63.3.8 int get_int_element (**set** ∗ *s,* int *key*)

4.63.3.9 **s_node**∗ new_s_node (int *key,* void ∗ *body*)

4.63.3.10 **set**∗ new_set (int *size*)

4.63.3.11 **set**∗ new_set_map (int *size*)

4.63.3.12 **set**∗ new_set_nomap (int *size*)

4.63.3.13 void∗ peek_element (**set** ∗ *s*)

4.63.3.14 void∗ remove_element (**set** ∗ *s*)

4.63.3.15 int remove_element_key (**set** ∗ *s*)

4.63.3.16 void∗ remove_element_with_key (**set** ∗ *s,* int *key,* int *get_keyvoid* ∗)

4.63.3.17 int remove_int_element (**set** ∗ *s,* int *key*)

4.63.3.18 void reset_set (**set** ∗ *s*)

4.63.3.19 int set_empty (**set** ∗ *s*)

4.63.3.20 **llist**∗ set_to_list (**set** ∗ *s*)

4.64 solution.c File Reference

```
#include "solution.h"
#include <stdio.h>
#include "utils.h"
#include "llist.h"
```

Functions

- solution ∗ read_solution (char ∗filename)
- int check_in_solution (htable ∗ht, int t, solution ∗sol)
- int table_has_the_solution (htable ∗ht, solution ∗sol)
- int assignment_has_the_solution (variables ∗vars, solution ∗sol)
- void confirm_variables (variables ∗vars, solution ∗sol)

4.64.1 Function Documentation

4.64.1.1 int assignment_has_the_solution (**variables** ∗ *vars,* **solution** ∗ *sol*)

4.64.1.2 int check_in_solution (**htable** ∗ *ht,* int *t,* **solution** ∗ *sol*)

4.64.1.3 void confirm_variables (**variables** ∗ *vars,* **solution** ∗ *sol*)

4.64.1.4 **solution**∗ read_solution (char ∗ *filename*)

4.64.1.5 int table_has_the_solution (**htable** ∗ *ht,* **solution** ∗ *sol*)

4.65 solution.h File Reference

```
#include "hashtable.h"
#include "variables.h"
```

Classes

- struct solution

Typedefs

- typedef struct solution solution

Functions

- solution ∗ read_solution (char ∗filename)
- int check_in_solution (htable ∗ht, int t, solution ∗sol)
- int table_has_the_solution (htable ∗ht, solution ∗sol)
- int assignment_has_the_solution (variables ∗vars, solution ∗sol)
- void confirm_variables (variables ∗vars, solution ∗sol)

4.65.1 Detailed Description

Represents a user-given solution and method for querying it. Mainly used for debugging.

4.65.2 Typedef Documentation

4.65.2.1 typedef struct solution solution

4.65.3 Function Documentation

4.65.3.1 int assignment_has_the_solution (**variables** ∗ *vars,* **solution** ∗ *sol*)

4.65.3.2 int check_in_solution (**htable** ∗ *ht,* int *t,* **solution** ∗ *sol*)

4.65.3.3 void confirm_variables (**variables** ∗ *vars,* **solution** ∗ *sol*)

4.65.3.4 **solution**∗ read_solution (char ∗ *filename*)

4.65.3.5 int table_has_the_solution (**htable** ∗ *ht,* **solution** ∗ *sol*)

4.66 sort.c File Reference

```
#include "utils.h"
#include "sort.h"
#include <stdlib.h>
#include <stdio.h>
#include "hashtable.h"
```

Functions

- void buble_sort_list (llist ∗tosort, int get_value(void ∗, void ∗), void ∗helper)
- void buble_sort_list_2 (llist ∗tosort, int compare(void ∗, void ∗))
- void buble_sort (int ∗a)
- void swap_int (int i, int j, void ∗a, int width, int get_value(void ∗, int, int), void set_value(void ∗, int, int, int))
- void swap_obj (int i, int j, void ∗a, int width, void ∗get_object(void ∗, int, int), void set_object(void ∗, int, int, void ∗))
- int is_sorted (void ∗a, int start, int end, int key_col, int get_value(void ∗, int, int))
- void sort_relative_to_col (void ∗a, int size, int width, int key_col, int key_col2, llist_node ∗cols, int get_value(void ∗, int, int), void ∗get_object(void ∗, int, int), void set_value(void ∗, int, int, int), void set_object(void ∗, int, int, void ∗), int setobject)
- void sort_w_obj_mcols (void ∗a, int size, int width, llist ∗cols, int get_value(void ∗, int, int), void ∗get_object(void ∗, int, int), void set_value(void ∗, int, int, int), void set_object(void ∗, int, int, void ∗), int setobject)
- void sort_w_obj (void ∗a, int size, int width, int key_col, int key_col2, int get_value(void ∗, int, int), void ∗get_object(void ∗, int, int), void set_value(void ∗, int, int, int), void set_object(void ∗, int, int, void ∗), int setobject)
- void sort_w_mcols (void ∗a, int size, int width, llist ∗cols, int get_value(void ∗, int, int), void set_value(void ∗, int, int, int))
- void sort_w (void ∗a, int size, int width, int key_col, int key_col2, int get_value(void ∗, int, int), void set_value(void ∗, int, int, int))
- int get_rand_value (void ∗av, int i, int c)
- void set_rand_value (void ∗av, int i, int c, int v)
- void sort_recursive (void ∗a, int p, int r, int key_col, int width, int randp, int get_value(void ∗, int, int), void ∗get_object(void ∗, int, int), void set_value(void ∗, int, int, int), void set_object(void ∗, int, int, void ∗), int setobject)

4.66.1 Function Documentation

4.66.1.1 void buble_sort (int ∗ *a*)

4.66.1.2 void buble_sort_list (llist ∗ *tosort,* int *get_valuevoid ∗, void ∗,* void ∗ *helper*)

4.66.1.3 void buble_sort_list_2 (llist ∗ *tosort,* int *comparevoid ∗, void ∗*)

4.66.1.4 int get_rand_value (void ∗ *av,* int *i,* int *c*)

4.66.1.5 int is_sorted (void ∗ *a,* int *start,* int *end,* int *key_col,* int *get_valuevoid ∗, int, int*)

4.66.1.6 void set_rand_value (void ∗ *av,* int *i,* int *c,* int *v*)

4.66.1.7 void sort_recursive (void ∗ *a,* int *p,* int *r,* int *key_col,* int *width,* int *randp,* int *get_valuevoid ∗, int, int,* void ∗ *get_objectvoid ∗, int, int,* void *set_valuevoid ∗, int, int, int,* void *set_objectvoid ∗, int, int, void ∗,* int *setobject*)

4.66.1.8 void sort_relative_to_col (void ∗ *a,* int *size,* int *width,* int *key_col,* int *key_col2,* llist_node ∗ *cols,* int *get_valuevoid ∗, int, int,* void ∗ *get_objectvoid ∗, int, int,* void *set_valuevoid ∗, int, int, int,* void *set_objectvoid ∗, int, int, void ∗,* int *setobject*)

4.66.1.9 void sort_w (void ∗ *a,* int *size,* int *width,* int *key_col,* int *key_col2,* int *get_valuevoid ∗, int, int,* void *set_valuevoid ∗, int, int, int*)

4.66.1.10 void sort_w_mcols (void ∗ *a,* int *size,* int *width,* llist ∗ *cols,* int *get_valuevoid ∗, int, int,* void *set_valuevoid ∗, int, int, int*)

4.66.1.11 void sort_w_obj (void ∗ *a,* int *size,* int *width,* int *key_col,* int *key_col2,* int *get_valuevoid ∗, int, int,* void ∗ *get_objectvoid ∗, int, int,* void *set_valuevoid ∗, int, int, int,* void *set_objectvoid ∗, int, int, void ∗,* int *setobject*)

4.66.1.12 void sort_w_obj_mcols (void ∗ *a,* int *size,* int *width,* llist ∗ *cols,* int *get_valuevoid* ∗*, int, int,* void ∗ *get_objectvoid* ∗*, int, int,* void *set_valuevoid* ∗*, int, int, int,* void *set_objectvoid* ∗*, int, int, void* ∗*,* int *setobject*)

4.66.1.13 void swap_int (int *i,* int *j,* void ∗ *a,* int *width,* int *get_valuevoid* ∗*, int, int,* void *set_valuevoid* ∗*, int, int, int*)

4.66.1.14 void swap_obj (int *i,* int *j,* void ∗ *a,* int *width,* void ∗ *get_objectvoid* ∗*, int, int,* void *set_objectvoid* ∗*, int, int, void* ∗)

4.67 sort.h File Reference

Functions

- void buble_sort_list (llist ∗tosort, int get_value(void ∗, void ∗), void ∗helper)
- void buble_sort_list_2 (llist ∗tosort, int compare(void ∗, void ∗))
- void buble_sort (int ∗a)
- void sort (void ∗a, int size, int key_col, int key_col2, int get_value(), void set_object(void ∗, int, int, void ∗))
- void sort_w (void ∗a, int size, int width, int key_col, int key_col2, int get_value(void ∗, int, int), void set_value(void ∗, int, int, int))
- void sort_relative_to_col (void ∗a, int size, int width, int key_col, int key_col2, llist_node ∗cols, int get_value(void ∗, int, int), void ∗get_object(void ∗, int, int), void set_value(void ∗, int, int, int), void set_object(void ∗, int, int, void ∗), int setobject)
- void sort_w_obj (void ∗a, int size, int width, int key_col, int key_col2, int get_value(void ∗, int, int), void ∗get_object(void ∗, int, int), void set_value(void ∗, int, int, int), void set_object(void ∗, int, int, void ∗), int setobject)
- int get_rand_value (void ∗av, int i, int c)
- void set_rand_value (void ∗av, int i, int c, int v)
- void sort_recursive (void ∗a, int p, int r, int key_col, int width, int randp, int get_value(void ∗, int, int), void ∗get_object(void ∗, int, int), void set_value(void ∗, int, int, int), void set_object(void ∗, int, int, void ∗), int setobject)
- void sort_w_obj_mcols (void ∗a, int size, int width, llist ∗cols, int get_value(void ∗, int, int), void ∗get_object(void ∗, int, int), void set_value(void ∗, int, int, int), void set_object(void ∗, int, int, void ∗), int setobject)
- void sort_w_mcols (void ∗a, int size, int width, llist ∗cols, int get_value(void ∗, int, int), void set_value(void ∗, int, int, int))
- void swap_int (int i, int j, void ∗a, int width, int get_value(void ∗, int, int), void set_value(void ∗, int, int, int))
- void swap_obj (int i, int j, void ∗a, int width, void ∗get_object(void ∗, int, int), void set_object(void ∗, int, int, void ∗))
- int is_sorted (void ∗a, int start, int end, int key_col, int get_value())

4.67.1 Detailed Description

Provides an implementation for quick-sort.

4.67.2 Function Documentation

4.67.2.1 void buble_sort (int ∗ *a*)

4.67.2.2 void buble_sort_list (llist ∗ *tosort,* int *get_valuevoid* ∗*, void* ∗*,* void ∗ *helper*)

4.67.2.3 void buble_sort_list_2 (llist ∗ *tosort,* int *comparevoid* ∗*, void* ∗)

4.67.2.4 int get_rand_value (void ∗ *av,* int *i,* int *c*)

4.67.2.5 int is_sorted (void ∗ *a,* int *start,* int *end,* int *key_col,* int *get_value()*)

4.67.2.6 void set_rand_value (void ∗ *av,* int *i,* int *c,* int *v*)

4.67.2.7 void sort (void ∗ *a,* int *size,* int *key_col,* int *key_col2,* int *get_value(),* void *set_objectvoid* ∗*, int, int, void* ∗)

4.67.2.8 void sort_recursive (void ∗ *a,* int *p,* int *r,* int *key_col,* int *width,* int *randp,* int *get_valuevoid* ∗*, int, int,* void ∗ *get_objectvoid* ∗*, int, int,* void *set_valuevoid* ∗*, int, int, int,* void *set_objectvoid* ∗*, int, int, void* ∗*,* int *setobject*)

4.67.2.9 void sort_relative_to_col (void ∗ *a,* int *size,* int *width,* int *key_col,* int *key_col2,* llist_node ∗ *cols,* int *get_valuevoid* ∗*, int, int,* void ∗ *get_objectvoid* ∗*, int, int,* void *set_valuevoid* ∗*, int, int, int,* void *set_objectvoid* ∗*, int, int, void* ∗*,* int *setobject*)

4.67.2.10 void sort_w (void ∗ *a,* int *size,* int *width,* int *key_col,* int *key_col2,* int *get_valuevoid* ∗*, int, int,* void *set_valuevoid* ∗*, int, int, int*)

4.67.2.11 void sort_w_mcols (void ∗ *a,* int *size,* int *width,* llist ∗ *cols,* int *get_valuevoid* ∗*, int, int,* void *set_valuevoid* ∗*, int, int, int*)

4.67.2.12 void sort_w_obj (void ∗ *a,* int *size,* int *width,* int *key_col,* int *key_col2,* int *get_valuevoid* ∗*, int, int,* void ∗ *get_objectvoid* ∗*, int, int,* void *set_valuevoid* ∗*, int, int, int,* void *set_objectvoid* ∗*, int, int, void* ∗*,* int *setobject*)

4.67.2.13 void sort_w_obj_mcols (void ∗ *a,* int *size,* int *width,* llist ∗ *cols,* int *get_valuevoid* ∗*, int, int,* void ∗ *get_objectvoid* ∗*, int, int,* void *set_valuevoid* ∗*, int, int, int,* void *set_objectvoid* ∗*, int, int, void* ∗*,* int *setobject*)

4.67.2.14 void swap_int (int *i,* int *j,* void ∗ *a,* int *width,* int *get_valuevoid* ∗*, int, int,* void *set_valuevoid* ∗*, int, int, int*)

4.67.2.15 void swap_obj (int *i,* int *j,* void ∗ *a,* int *width,* void ∗ *get_objectvoid* ∗*, int, int,* void *set_objectvoid* ∗*, int, int, void* ∗)

4.68 tree_map.c File Reference

```
#include "tree_map.h"
```

Functions

- tree_map ∗ new_tree_map (htable ∗htto, llist ∗common_cols, int skip_deleted)
- void compress_tree_map (tree_map ∗tm)
- tree_map_node ∗ compress_tree_map_rec (llist ∗ll, llist_node ∗level)
- int ∗∗ compress_tuple_list (llist ∗ll)
- tree_map_node ∗ compress_tree_map_node_list (llist ∗ll)
- tree_map_node ∗ find_node_with_id_compressesd (int id, tree_map_node ∗al, int size, int search, int wildcard)
- tree_map_node ∗ find_node_with_id (int id, llist ∗l, int search, int wildcard)
- tree_map_node ∗ new_tree_map_node (int id)
- int get_tuples_for_tuple_compressed (int ∗tuple, tree_map ∗tm, llist ∗common_cols, int wildcard, int ∗∗∗result)
- llist ∗ get_tuples_for_tuple (int ∗tuple, tree_map ∗tm, llist ∗common_cols, int wildcard)
- int insert_tuple_into_tree (int ∗tuple, tree_map ∗tm, llist ∗common_cols, int wildcard)
- int insert_tuple_into_tree_payload (int ∗tuple, tree_map ∗tm, llist ∗common_cols, int wildcard, void ∗payload)
- htable ∗ tree_map_to_htable_rec (tree_map ∗tm, char ∗name, int id, char ∗∗var_names, int ∗vars)
- void fetch_tuples (tree_map_node ∗tmn, htable ∗ht, int ∗tuple, int depth, int length)
- void reset_tree_map (tree_map ∗tm)
- void destroy_tree_map (tree_map ∗tm, int payload)
- void destroy_tree_map_lists (llist ∗l, int level, int limit, int destroy_payload)
- void destroy_tree_map_arrays (tree_map_node ∗la, int level, int limit, int destroy_payload, int size)
- htable ∗ tree_map_to_htable (tree_map ∗tm, char ∗name, int id, char ∗∗var_names, int ∗vars, htable ∗tofill)
- void traverse_treemap (tree_map ∗tm)

4.68.1 Function Documentation

4.68.1.1 void compress_tree_map (**tree_map** ∗ *tm*)

4.68.1.2 **tree_map_node**∗ compress_tree_map_node_list (**llist** ∗ *ll*)

4.68.1.3 **tree_map_node**∗ compress_tree_map_rec (**llist** ∗ *ll*, **llist_node** ∗ *level*)

4.68.1.4 int∗∗ compress_tuple_list (**llist** ∗ *ll*)

4.68.1.5 void destroy_tree_map (**tree_map** ∗ *tm*, int *payload*)

4.68.1.6 void destroy_tree_map_arrays (**tree_map_node** ∗ *la*, int *level*, int *limit*, int *destroy_payload*, int *size*)

4.68.1.7 void destroy_tree_map_lists (**llist** ∗ *l*, int *level*, int *limit*, int *destroy_payload*)

4.68.1.8 void fetch_tuples (**tree_map_node** ∗ *tmn*, **htable** ∗ *ht*, int ∗ *tuple*, int *depth*, int *length*)

4.68.1.9 **tree_map_node**∗ find_node_with_id (int *id*, **llist** ∗ *l*, int *search*, int *wildcard*)

4.68.1.10 **tree_map_node**∗ find_node_with_id_compressed (int *id*, **tree_map_node** ∗ *al*, int *size*, int *search*, int *wildcard*)

4.68.1.11 **llist**∗ get_tuples_for_tuple (int ∗ *tuple*, **tree_map** ∗ *tm*, **llist** ∗ *common_cols*, int *wildcard*)

4.68.1.12 int get_tuples_for_tuple_compressed (int ∗ *tuple*, **tree_map** ∗ *tm*, **llist** ∗ *common_cols*, int *wildcard*, int ∗∗∗ *result*)

4.68.1.13 int insert_tuple_into_tree (int ∗ *tuple*, **tree_map** ∗ *tm*, **llist** ∗ *common_cols*, int *wildcard*)

4.68.1.14 int insert_tuple_into_tree_payload (int ∗ *tuple*, **tree_map** ∗ *tm*, **llist** ∗ *common_cols*, int *wildcard*, void ∗ *payload*)

4.68.1.15 **tree_map**∗ new_tree_map (**htable** ∗ *htto*, **llist** ∗ *common_cols*, int *skip_deleted*)

4.68.1.16 **tree_map_node**∗ new_tree_map_node (int *id*)

4.68.1.17 void reset_tree_map (**tree_map** ∗ *tm*)

4.68.1.18 void traverse_treemap (**tree_map** ∗ *tm*)

4.68.1.19 **htable**∗ tree_map_to_htable (**tree_map** ∗ *tm*, char ∗ *name*, int *id*, char ∗∗ *var_names*, int ∗ *vars*, **htable** ∗ *tofill*)

4.68.1.20 **htable**∗ tree_map_to_htable_rec (**tree_map** ∗ *tm*, char ∗ *name*, int *id*, char ∗∗ *var_names*, int ∗ *vars*)

4.69 tree_map.h File Reference

```
#include "utils.h"
#include "hashtable.h"
#include "llist.h"
```

Classes

- struct tree_map
- struct tree_map_node

Macros

- #define STAR -99

Typedefs

- typedef struct tree_map tree_map
- typedef struct tree_map_node tree_map_node

Functions

- tree_map ∗ new_tree_map (htable ∗htto, llist ∗common_cols, int skip_deleted)
- tree_map_node ∗ find_node_with_id (int id, llist ∗l, int search, int wildcard)
- tree_map_node ∗ find_node_with_id_compressesd (int id, tree_map_node ∗al, int size, int search, int wildcard)
- tree_map_node ∗ new_tree_map_node (int id)
- llist ∗ get_tuples_for_tuple (int ∗tuple, tree_map ∗tm, llist ∗common_cols, int wildcard)
- int get_tuples_for_tuple_compressed (int ∗tuple, tree_map ∗tm, llist ∗common_cols, int wildcard, int ∗∗∗result)
- int insert_tuple_into_tree_payload (int ∗tuple, tree_map ∗tm, llist ∗common_cols, int wildcard, void ∗payload)
- int insert_tuple_into_tree (int ∗tuple, tree_map ∗tm, llist ∗common_cols, int wildcard)
- htable ∗ tree_map_to_htable (tree_map ∗tm, char ∗name, int id, char ∗∗var_names, int ∗vars, htable ∗tofill)
- htable ∗ tree_map_to_htable_rec (tree_map ∗tm, char ∗name, int id, char ∗∗var_names, int ∗vars)
- void fetch_tuples (tree_map_node ∗tmn, htable ∗ht, int ∗tuple, int depth, int length)
- void destroy_tree_map (tree_map ∗tm, int payload)
- void destroy_tree_map_lists (llist ∗l, int level, int limit, int destroy_payload)
- void destroy_tree_map_arrays (tree_map_node ∗la, int level, int limit, int destroy_payload, int size)
- void traverse_treemap (tree_map ∗tm)
- void reset_tree_map (tree_map ∗tm)
- tree_map_node ∗ compress_tree_map_rec (llist ∗ll, llist_node ∗level)
- void compress_tree_map (tree_map ∗tm)
- int ∗∗ compress_tuple_list (llist ∗ll)
- tree_map_node ∗ compress_tree_map_node_list (llist ∗ll)

4.69.1 Detailed Description

Implements the index-tree data structure for finding matching tuples in relations. Also provides compression algorithms.

4.69.2 Macro Definition Documentation

4.69.2.1 #define STAR -99

4.69.3 Typedef Documentation

4.69.3.1 typedef struct tree_map tree_map

4.69.3.2 typedef struct tree_map_node tree_map_node

4.69.4 Function Documentation

4.69.4.1 void compress_tree_map (tree_map ∗ *tm*)

4.69.4.2 tree_map_node∗ compress_tree_map_node_list (llist ∗ *ll*)

4.69.4.3 tree_map_node∗ compress_tree_map_rec (llist ∗ *ll,* llist_node ∗ *level*)

4.69.4.4 int∗∗ compress_tuple_list (llist ∗ *ll*)

4.69.4.5 void destroy_tree_map (tree_map ∗ *tm,* int *payload*)

4.69.4.6 void destroy_tree_map_arrays (tree_map_node ∗ *la,* int *level,* int *limit,* int *destroy_payload,* int *size*)

4.69.4.7 void destroy_tree_map_lists (llist ∗ *l,* int *level,* int *limit,* int *destroy_payload*)

4.69.4.8 void fetch_tuples (tree_map_node ∗ *tmn,* htable ∗ *ht,* int ∗ *tuple,* int *depth,* int *length*)

4.69.4.9 tree_map_node∗ find_node_with_id (int *id,* llist ∗ *l,* int *search,* int *wildcard*)

4.69.4.10 tree_map_node∗ find_node_with_id_compressesd (int *id,* tree_map_node ∗ *al,* int *size,* int *search,* int *wildcard*)

4.69.4.11 llist∗ get_tuples_for_tuple (int ∗ *tuple,* tree_map ∗ *tm,* llist ∗ *common_cols,* int *wildcard*)

4.69.4.12 int get_tuples_for_tuple_compressed (int ∗ *tuple,* tree_map ∗ *tm,* llist ∗ *common_cols,* int *wildcard,* int ∗∗∗ *result*)

4.69.4.13 int insert_tuple_into_tree (int ∗ *tuple,* tree_map ∗ *tm,* llist ∗ *common_cols,* int *wildcard*)

4.69.4.14 int insert_tuple_into_tree_payload (int ∗ *tuple,* tree_map ∗ *tm,* llist ∗ *common_cols,* int *wildcard,* void ∗ *payload*)

4.69.4.15 tree_map∗ new_tree_map (htable ∗ *htto,* llist ∗ *common_cols,* int *skip_deleted*)

4.69.4.16 tree_map_node∗ new_tree_map_node (int *id*)

4.69.4.17 void reset_tree_map (tree_map ∗ *tm*)

4.69.4.18 void traverse_treemap (tree_map ∗ *tm*)

4.69.4.19 htable∗ tree_map_to_htable (tree_map ∗ *tm,* char ∗ *name,* int *id,* char ∗∗ *var_names,* int ∗ *vars,* htable ∗ *tofill*)

4.69.4.20 htable∗ tree_map_to_htable_rec (tree_map ∗ *tm,* char ∗ *name,* int *id,* char ∗∗ *var_names,* int ∗ *vars*)

4.70 utils.c File Reference

```
#include <execinfo.h>
#include <malloc.h>
#include "utils.h"
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "llist.h"
#include "globals.h"
```

Macros

- #define PROFILE_SIZE 2000

Functions

- void profile_time_start (int id, const char ∗desc)
- double ∗ get_profile_times (void)
- void profile_time_end (int id)
- void profile_crash_all_end (void)
- void print_profile (void)
- int string_equal_casted (void ∗a, void ∗b)
- int string_equal (char ∗a, char ∗b)
- char ∗ str_copy (char ∗a)
- int compare_ints (const void ∗a, const void ∗b)
- char ∗ str_copy_short (char ∗a, int max)
- int max (int a, int b)
- char ∗ str_cat (char ∗a, const char ∗b)
- char ∗ str_cat_sep (char ∗a, char ∗b, char ∗s)
- void warn (const char ∗a)
- void error (const char ∗a)
- char ∗ str_cat_recursive (char ∗a, char ∗b, const char ∗s)
- void ∗ checked_malloc (int size)
- void checked_free (void ∗buffer)
- int skip_char (char ∗line, int i, char c)
- int skip_space (char ∗line, int i)
- int get_int (char ∗line, int i, int ∗num)
- int get_double (char ∗line, int i, double ∗num)
- int skip_to_char (char ∗line, int i, char c)
- int get_string (char ∗line, int i, char ∗∗str)
- int visit_var (int var)
- void unvisit_var (int var)
- int set_var (int var)
- void reset_vars (void)
- void print_trace (void)
- void catcher (int sig)
- void dump_trace (void)

Variables

- double profiled [PROFILE_SIZE]
- char ∗ profile_desc [PROFILE_SIZE]
- int profiled_count [PROFILE_SIZE]

4.70.1 Macro Definition Documentation

4.70.1.1 #define PROFILE_SIZE 2000

4.70.2 Function Documentation

4.70.2.1 void catcher (int *sig*)

4.70.2.2 void checked_free (void ∗ *buffer*)

4.70.2.3 void∗ checked_malloc (int *size*)

4.70.2.4 int compare_ints (const void ∗ *a,* const void ∗ *b*)

4.70.2.5 void dump_trace (void)

4.70.2.6 void error (const char ∗ *a*)

4.70.2.7 int get_double (char ∗ *line,* int *i,* double ∗ *num*)

4.70.2.8 int get_int (char ∗ *line,* int *i,* int ∗ *num*)

4.70.2.9 double∗ get_profile_times (void)

4.70.2.10 int get_string (char ∗ *line,* int *i,* char ∗∗ *str*)

4.70.2.11 int max (int *a,* int *b*)

4.70.2.12 void print_profile (void)

4.70.2.13 void print_trace (void)

4.70.2.14 void profile_crash_all_end (void)

4.70.2.15 void profile_time_end (int *id*)

4.70.2.16 void profile_time_start (int *id,* const char ∗ *desc*)

4.70.2.17 void reset_vars (void)

4.70.2.18 int set_var (int *var*)

4.70.2.19 int skip_char (char ∗ *line,* int *i,* char *c*)

4.70.2.20 int skip_space (char ∗ *line,* int *i*)

4.70.2.21 int skip_to_char (char ∗ *line,* int *i,* char *c*)

4.70.2.22 char∗ str_cat (char ∗ *a,* const char ∗ *b*)

4.70.2.23 char∗ str_cat_recursive (char ∗ *a,* char ∗ *b,* const char ∗ *s*)

4.70.2.24 char∗ str_cat_sep (char ∗ *a,* char ∗ *b,* char ∗ *s*)

4.70.2.25 char∗ str_copy (char ∗ *a*)

4.70.2.26 char∗ str_copy_short (char ∗ *a,* int *max*)

4.70.2.27 int string_equal (char ∗ *a,* char ∗ *b*)

4.70.2.28 int string_equal_casted (void ∗ *a,* void ∗ *b*)

4.70.2.29 void unvisit_var (int *var*)

4.70.2.30 int visit_var (int *var*)

4.70.2.31 void warn (const char ∗ *a*)

4.70.3 Variable Documentation

4.70.3.1 char∗ profile_desc[PROFILE_SIZE]

4.70.3.2 double profiled[PROFILE_SIZE]

4.70.3.3 int profiled_count[PROFILE_SIZE]

4.71 utils.h File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include "globals.h"
```

Functions

- void profile_time_start (int id, const char ∗desc)
- double ∗ get_profile_times (void)
- void profile_time_end (int id)
- void print_profile (void)
- int string_equal_casted (void ∗a, void ∗b)
- int string_equal (char ∗a, char ∗b)
- char ∗ str_copy (char ∗a)
- char ∗ str_copy_short (char ∗a, int max)
- char ∗ str_cat (char ∗a, const char ∗b)
- char ∗ str_cat_sep (char ∗a, char ∗b, char ∗s)
- char ∗ str_cat_recursive (char ∗a, char ∗b, const char ∗s)
- void warn (const char ∗a)
- void error (const char ∗a)
- void ∗ checked_malloc (int size)
- void checked_free (void ∗buffer)
- int max (int a, int b)
- int compare_ints (const void ∗a, const void ∗b)
- int skip_char (char ∗line, int i, char c)
- int skip_space (char ∗line, int i)
- int get_int (char ∗line, int i, int ∗num)
- int get_double (char ∗line, int i, double ∗num)
- int skip_to_char (char ∗line, int i, char c)
- int get_string (char ∗line, int i, char ∗∗str)
- int set_var (int var)
- void reset_vars (void)
- int visit_var (int var)
- void unvisit_var (int var)
- void print_trace (void)
- void print_allocs (void)
- void profile_crash_all_end (void)
- void catcher (int sig)
- void dump_trace (void)

4.71.1 Detailed Description

Implements various utilities, such as code profiling tools.

4.71.2 Function Documentation

4.71.2.1 void catcher (int *sig*)

4.71.2.2 void checked_free (void ∗ *buffer*)

4.71.2.3 void∗ checked_malloc (int *size*)

4.71.2.4 int compare_ints (const void ∗ *a,* const void ∗ *b*)

4.71.2.5 void dump_trace (void)

4.71.2.6 void error (const char ∗ *a*)

4.71.2.7 int get_double (char ∗ *line,* int *i,* double ∗ *num*)

4.71.2.8 int get_int (char ∗ *line,* int *i,* int ∗ *num*)

4.71.2.9 double∗ get_profile_times (void)

4.71.2.10 int get_string (char ∗ *line,* int *i,* char ∗∗ *str*)

4.71.2.11 int max (int *a,* int *b*)

4.71.2.12 void print_allocs (void)

4.71.2.13 void print_profile (void)

4.71.2.14 void print_trace (void)

4.71.2.15 void profile_crash_all_end (void)

4.71.2.16 void profile_time_end (int *id*)

4.71.2.17 void profile_time_start (int *id,* const char ∗ *desc*)

4.71.2.18 void reset_vars (void)

4.71.2.19 int set_var (int *var*)

4.71.2.20 int skip_char (char ∗ *line,* int *i,* char *c*)

4.71.2.21 int skip_space (char ∗ *line,* int *i*)

4.71.2.22 int skip_to_char (char ∗ *line,* int *i,* char *c*)

4.71.2.23 char∗ str_cat (char ∗ *a,* const char ∗ *b*)

4.71.2.24 char∗ str_cat_recursive (char ∗ *a,* char ∗ *b,* const char ∗ *s*)

4.71.2.25 char∗ str_cat_sep (char ∗ *a,* char ∗ *b,* char ∗ *s*)

4.71.2.26 char∗ str_copy (char ∗ *a*)

4.71.2.27 char∗ str_copy_short (char ∗ *a,* int *max*)

4.71.2.28 int string_equal (char ∗ *a,* char ∗ *b*)

4.71.2.29 int string_equal_casted (void ∗ *a,* void ∗ *b*)

4.71.2.30 void unvisit_var (int *var*)

4.71.2.31 int visit_var (int *var*)

4.71.2.32 void warn (const char ∗ *a*)

4.72 variables.c File Reference

```
#include <stdio.h>
#include "variables.h"
#include "utils.h"
#include "llist.h"
#include "domains.h"
#include "rb_tree.h"
```

Functions

- var_val ∗ new_var_val (void)
- variables ∗ init_variables (int count)
- int set_value (variables ∗vars, int var, int val)
- void add_variable (llist ∗domain, variables ∗vars, int id, const char ∗name)
- int no_solutions (variables ∗vars)
- void print_assigned_vars (variables ∗vars)
- void print_solution (variables ∗vars)
- int compare_variable_number (void ∗value, void ∗bodyb)
- int get_variable_number (void ∗var, void ∗helper)
- void print_variables (variables ∗vars, const char ∗notes, int dom)
- void print_vars_for_sol_checker (variables ∗vars, char ∗name)
- void set_static_variable_ordering (variables ∗vars, char ∗filename)
- void init_domain_sets (variables ∗vars)
- void print_variable_domain_size (variables ∗vars, int id)
- void reset_variable_constraints (variables ∗vars)
- int scope_contained_in (int ∗scopea, llist ∗scopeal, int ∗scopeb, llist ∗scopebl, int ∗varmap, set ∗projected_on)

  if b is inside a then 1

4.72.1 Function Documentation

4.72.1.1 void add_variable (**llist** ∗ *domain,* **variables** ∗ *vars,* int *id,* const char ∗ *name*)

4.72.1.2 int compare_variable_number (void ∗ *value,* void ∗ *bodyb*)

4.72.1.3 int get_variable_number (void ∗ *var,* void ∗ *helper*)

4.72.1.4 void init_domain_sets (**variables** ∗ *vars*)

4.72.1.5 **variables**∗ init_variables (int *count*)

4.72.1.6 var_val∗ new_var_val (void)

4.72.1.7 int no_solutions (variables ∗ *vars*)

4.72.1.8 void print_assigned_vars (variables ∗ *vars*)

4.72.1.9 void print_solution (variables ∗ *vars*)

4.72.1.10 void print_variable_domain_size (variables ∗ *vars,* int *id*)

4.72.1.11 void print_variables (variables ∗ *vars,* const char ∗ *notes,* int *dom*)

4.72.1.12 void print_vars_for_sol_checker (variables ∗ *vars,* char ∗ *name*)

4.72.1.13 void reset_variable_constraints (variables ∗ *vars*)

4.72.1.14 int scope_contained_in (int ∗ *scopea,* llist ∗ *scopeal,* int ∗ *scopeb,* llist ∗ *scopebl,* int ∗ *varmap,* set ∗ *projected_on*)

if b is inside a then 1

4.72.1.15 void set_static_variable_ordering (variables ∗ *vars,* char ∗ *filename*)

4.72.1.16 int set_value (variables ∗ *vars,* int *var,* int *val*)

4.73 variables.h File Reference

```
#include <stdlib.h>
#include "llist.h"
#include "lstack.h"
#include "domains.h"
#include "set.h"
```

Classes

- struct variable
- struct variables
- struct var_val

Typedefs

- typedef struct variable variable
- typedef struct variables variables
- typedef struct var_val var_val

Functions

- var_val ∗ new_var_val (void)
- int set_value (variables ∗vars, int var, int val)
- variables ∗ init_variables (int count)
- void add_variable (llist ∗domain, variables ∗vars, int id, const char ∗name)
- int compare_variable_number (void ∗value, void ∗bodyb)

- int get_variable_number (void ∗var, void ∗helper)
- int no_solutions (variables ∗vars)
- void print_variables (variables ∗vars, const char ∗notes, int dom)
- void print_solution (variables ∗vars)
- void print_vars_for_sol_checker (variables ∗vars, char ∗name)
- void set_neighbours (variables ∗vars)
- void init_vars (variables ∗vars)
- void set_static_variable_ordering (variables ∗vars, char ∗filename)
- void init_domain_sets (variables ∗vars)
- void print_variable_domain_size (variables ∗vars, int id)
- void print_assigned_vars (variables ∗vars)
- void reset_variable_constraints (variables ∗vars)
- int scope_contained_in (int ∗scopea, llist ∗scopeal, int ∗scopeb, llist ∗scopebl, int ∗varmap, set ∗projected_on)

  if b is inside a then 1

4.73.1 Detailed Description

Provides the data structures for representing the CSP variables and methods for constructing and manipulating them.

4.73.2 Typedef Documentation

4.73.2.1 typedef struct var_val var_val

4.73.2.2 typedef struct variable variable

4.73.2.3 typedef struct variables variables

4.73.3 Function Documentation

4.73.3.1 void add_variable (llist ∗ *domain,* variables ∗ *vars,* int *id,* const char ∗ *name*)

4.73.3.2 int compare_variable_number (void ∗ *value,* void ∗ *bodyb*)

4.73.3.3 int get_variable_number (void ∗ *var,* void ∗ *helper*)

4.73.3.4 void init_domain_sets (variables ∗ *vars*)

4.73.3.5 variables∗ init_variables (int *count*)

4.73.3.6 void init_vars (variables ∗ *vars*)

4.73.3.7 var_val∗ new_var_val (void)

4.73.3.8 int no_solutions (variables ∗ *vars*)

4.73.3.9 void print_assigned_vars (variables ∗ *vars*)

4.73.3.10 void print_solution (variables ∗ *vars*)

4.73.3.11 void print_variable_domain_size (variables ∗ *vars,* int *id*)

4.73.3.12 void print_variables (variables ∗ *vars,* const char ∗ *notes,* int *dom*)

4.73.3.13 void print_vars_for_sol_checker (variables ∗ *vars,* char ∗ *name*)

4.73.3.14 void reset_variable_constraints (variables ∗ *vars*)

4.73.3.15 int scope_contained_in (int ∗ *scopea,* llist ∗ *scopeal,* int ∗ *scopeb,* llist ∗ *scopebl,* int ∗ *varmap,* set ∗ *projected_on*)

if b is inside a then 1

4.73.3.16 void set_neighbours (variables ∗ *vars*)

4.73.3.17 void set_static_variable_ordering (variables ∗ *vars,* char ∗ *filename*)

4.73.3.18 int set_value (variables ∗ *vars,* int *var,* int *val*)