graphtools Documentation

Release 1.0

Gautam Sisodia

September 25, 2014

CONTENTS

CHAPTER

ONE

GRAPHTOOLS PACKAGE

1.1 graphtools Package

Hello peeps

1.2 gengraph Module

class graphtools.gengraph.**GenGraph**
A general graph.

count_neighbors(*vert*, *out=True*, *cond=False*, *less=True*, *cutoff=0*)
Count the number of neighbors of a vertex.

Parameters

- **vert** (*vertex*) – the vertex to count the neighbors of
- **out** (*Boolean*) – if True, count out neighbors, else in
- **cond** (*Boolean*) – if True, count the neighbors satisfying a condition on rank
- **less** (*Boolean*) – if True, count the neighbors with rank less than or equal to the cutoff, else more
- **cutoff** (*int*) – the cutoff rank for the conditional

descend(*vert*, *debug=False*)
Run one iteration of the descend algorithm.

vert: the vertex whose rank may change

descent(*num=1*, *debug=False*)
Run descend num times on random vertices.

get_num_arrows()
Return the number of arrows.

get_rank(*vert*)
Return the rank of vertex vert.

get_vert_list()
Return a list of vertices.

set_rank(*vert*, *newrank*)
Set the rank of vertex vert to int newrank.

1.3 dbgraph Module

class graphtools.dbgraph.**DBGraph**(*users*, *arrows*, *conn*, *group=None*)
Bases: graphtools.gengraph.GenGraph

A subclass of GenGraph for graphs stored in databases.

The vertices are stored in a table called users. The arrows are stored in a table called arrows. The vertices are identified by the value in the user_id column.

check_id(*vert*)

count_neighbors(*vert*, *out=True*, *cond=False*, *less=True*, *cutoff=0*)
Return the number of neighbors of a vertex.

vert: a vertex; count this vertex's neighbors

out: a Boolean; if True, count out neighbors, else in

cond: a Boolean; if True, count the neighbors satisfying a condition on rank

less: a Boolean; if True, count the neighbors with rank less than or equal to the cutoff, else more

cutoff: an int; the cutoff rank for the conditional

get_num_arrows()
Return the number of arrows.

get_rank(*vert*)
Return the rank of vertex vert.

get_vert_list()
return the list of user_ids.

reset_ranks()
Set all ranks to zero.

set_rank(*vert*, *newrank*)
Set the rank of vertex vert to int newrank.

1.4 listgraph Module

class graphtools.listgraph.**ListGraph**(*arrows_list*)
Bases: graphtools.gengraph.GenGraph

A subclass of GenGraph for graphs given as a list of arrows.

count_neighbors(*vert*, *out=True*, *cond=False*, *less=True*, *cutoff=0*)
Return the number of neighbors of a vertex.

vert: a vertex; count this vertex's neighbors

out: a Boolean; if True, count out neighbors, else in

cond: a Boolean; if True, count the neighbors satisfying a condition on rank

less: a Boolean; if True, count the neighbors with rank less than or equal to the cutoff, else more

cutoff: an int; the cutoff rank for the conditional

get_num_arrows()
Return the number of arrows.

get_rank(*vert*)
Return the rank of vertex vert.

get_vert_list()
Return a list of vertices.

neighbors_in(*vert*)
return the list of in neighbors of vertex vert.

neighbors_out(*vert*)
return the list of out neighbors of vertex vert.

rankdict = None
The rank dictionary; keys are the vertices and values are the ranks.

set_rank(*vert*, *newrank*)
Set the rank of vertex vert to int newrank.

1.5 sagegraph Module

class graphtools.sagegraph.**SageGraph**(*dg*)
Bases: graphtools.gengraph.GenGraph

A subclass of GenGraph which wraps a Sage DiGraph object.

count_neighbors(*vert*, *out=True*, *cond=False*, *less=True*, *cutoff=0*)
Return the number of neighbors of a vertex.

vert: a vertex; count this vertex's neighbors

out: a Boolean; if True, count out neighbors, else in

cond: a Boolean; if True, count the neighbors satisfying a condition on rank

less: a Boolean; if True, count the neighbors with rank less than or equal to the cutoff, else more

cutoff: an int; the cutoff rank for the conditional

get_num_arrows()
Return the number of arrows.

get_rank(*vert*)
Return the rank of vertex vert.

get_vert_list()
Return a list of vertices.

rankdict = None
The rank dictionary; keys are the vertices and values are the ranks.

set_rank(*vert*, *newrank*)
Set the rank of vertex vert to int newrank.

CHAPTER

TWO

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

PYTHON MODULE INDEX

g

INDEX