

---

# **graphtools Documentation**

***Release***

**Author**

September 25, 2014



# CONTENTS

|          |                              |          |
|----------|------------------------------|----------|
| <b>1</b> | <b>graphtools Package</b>    | <b>1</b> |
| 1.1      | graphtools Package . . . . . | 1        |
| 1.2      | dbgraph Module . . . . .     | 1        |
| 1.3      | gengraph Module . . . . .    | 2        |
| 1.4      | listgraph Module . . . . .   | 2        |
| 1.5      | privatedb Module . . . . .   | 3        |
| 1.6      | sagegraph Module . . . . .   | 3        |
| <b>2</b> | <b>Indices and tables</b>    | <b>5</b> |
|          | <b>Python Module Index</b>   | <b>7</b> |
|          | <b>Index</b>                 | <b>9</b> |



# GRAPHTOOLS PACKAGE

## 1.1 graphtools Package

Hello peeps

## 1.2 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.

## 1.3 gengraph Module

Created on Mon Sep 15 21:58:23 2014

@author: gautam

**class** `graphtools.gengraph.GenGraph`

A general graph.

**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

**descend** (*vert, debug=False*)

Run one iteration of the descend algorithm.

vert: the vertex whose rank may change

**descent** (*num=1, debug=False*)

Run descent 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.

## 1.4 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 privatedb Module

## 1.6 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*.





# INDICES AND TABLES

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



# PYTHON MODULE INDEX

## g

- `graphtools.__init__`, 1
- `graphtools.dbgraph`, 1
- `graphtools.gengraph`, 2
- `graphtools.listgraph`, 2
- `graphtools.privatedb`, 3
- `graphtools.sagegraph`, 3



# INDEX

## C

check\_id() (graphtools.dbgraph.DBGraph method), 1  
count\_neighbors() (graphtools.dbgraph.DBGraph method), 1  
count\_neighbors() (graphtools.gengraph.GenGraph method), 2  
count\_neighbors() (graphtools.listgraph.ListGraph method), 2  
count\_neighbors() (graphtools.sagegraph.SageGraph method), 3

## D

DBGraph (class in graphtools.dbgraph), 1  
descend() (graphtools.gengraph.GenGraph method), 2  
descent() (graphtools.gengraph.GenGraph method), 2

## G

GenGraph (class in graphtools.gengraph), 2  
get\_num\_arrows() (graphtools.dbgraph.DBGraph method), 1  
get\_num\_arrows() (graphtools.gengraph.GenGraph method), 2  
get\_num\_arrows() (graphtools.listgraph.ListGraph method), 2  
get\_num\_arrows() (graphtools.sagegraph.SageGraph method), 3  
get\_rank() (graphtools.dbgraph.DBGraph method), 1  
get\_rank() (graphtools.gengraph.GenGraph method), 2  
get\_rank() (graphtools.listgraph.ListGraph method), 3  
get\_rank() (graphtools.sagegraph.SageGraph method), 3  
get\_vert\_list() (graphtools.dbgraph.DBGraph method), 1  
get\_vert\_list() (graphtools.gengraph.GenGraph method), 2  
get\_vert\_list() (graphtools.listgraph.ListGraph method), 3  
get\_vert\_list() (graphtools.sagegraph.SageGraph method), 3  
graphtools.\_\_init\_\_ (module), 1  
graphtools.dbgraph (module), 1  
graphtools.gengraph (module), 2  
graphtools.listgraph (module), 2  
graphtools.privatedb (module), 3

graphtools.sagegraph (module), 3

## L

ListGraph (class in graphtools.listgraph), 2

## N

neighbors\_in() (graphtools.listgraph.ListGraph method), 3  
neighbors\_out() (graphtools.listgraph.ListGraph method), 3

## R

rankdict (graphtools.listgraph.ListGraph attribute), 3  
rankdict (graphtools.sagegraph.SageGraph attribute), 3  
reset\_ranks() (graphtools.dbgraph.DBGraph method), 1

## S

SageGraph (class in graphtools.sagegraph), 3  
set\_rank() (graphtools.dbgraph.DBGraph method), 1  
set\_rank() (graphtools.gengraph.GenGraph method), 2  
set\_rank() (graphtools.listgraph.ListGraph method), 3  
set\_rank() (graphtools.sagegraph.SageGraph method), 3