Python Webgraph Generator Reference Manual

Generated by Doxygen 1.4.6

Tue Jul 24 19:51:55 2007

Contents

Chapter 1

Python Webgraph Generator Main Page

A threaded Web graph (Power law random graph) generator written in Python. It can generate a synthetic Web graph of about one million nodes in a few minutes on a desktop machine. It implements a threaded variant of the RMAT algorithm. A little tweak can produce graphs representing social-networks or community-networks

Chapter 2

Python Webgraph Generator Namespace Index

2.1 Python Webgraph Generator Namespace List

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

Chapter 3

Python Webgraph Generator Hierarchical Index

3.1 Python Webgraph Generator Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

Python Webgraph Generator Class Index

4.1 Python Webgraph Generator Class List

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

Chapter 5

Python Webgraph Generator Page Index

5.1 Python Webgraph Generator Related Pages

Here is a list of all related documentation pages:

Chapter 6

Python Webgraph Generator Namespace Documentation

6.1 BaseElements Namespace Reference

6.1.1 Detailed Description

A module representing basic graph elements.

Classes

- class Vertex

  Represents graph vertex.

- class WeightedVertex

  Represents a weighted vertex.

- class WeightedVertices

  Represents a collection of weighted vertices of type BaseElements::WeightedVertices.

- class Edge

  Represents graph edge.

6.2 ChooseEdges Namespace Reference

6.2.1 Detailed Description

Threaded edge selection implementation.

Classes

- class ChooseEdges

  Thread for selecting a set of edges.

6.3 Graph Namespace Reference

6.3.1 Detailed Description

A module for representing graphs.

Classes

- class NumberedEdgeGraph

  Represents a numbered edge graph.

6.4 PackageExceptions Namespace Reference

6.4.1 Detailed Description

A module handling package exceptions.

Classes

- class Error

  Empty base class from which all exceptions are derived.

- class VertexError

  Represents a VertexError exception.

- class EdgeError

  Represents a EdgeError exception.

- class DistError

  Represents a DistError exception.

- class ErrorMessages

  Collection of various error message strings.

6.5 RandomGraphs Namespace Reference

6.5.1 Detailed Description

Collection of random graph generation/manipulation algorithms.

Currently the collection has just one algorithm ;-)

Classes

- class PowerLawRandomGraph

  Generates a synthetic Web graph or Power Law graph using an RMAT algorithm.

Chapter 7

Python Webgraph Generator Class Documentation

7.1 ChooseEdges::ChooseEdges Class Reference

7.1.1 Detailed Description

Thread for selecting a set of edges.

Public Member Functions

- def __init__
  Constructs a selector thread.
- def selectVertex
  Selects start and end vertices recursively.
- def run
  Start the thread.

Public Attributes

- startVertX
- endVertX
- startVertY
- endVertY
- noOfEdges
- probA
- probB
- probC
- probD
- **debug**
- id

Thread ID.

Static Public Attributes

- list serialEdgeList = [ ]

  Common serial edge list.

- tuple serialEdgeListLock = threading.Lock()

  Lock that a thread acquires for performing a semaphoric operation.

- int id = 0

  Thread ID.

7.1.2 Member Function Documentation

7.1.2.1 def ChooseEdges::ChooseEdges::__init__ (*self*, *noOfEdges*, *startVertX*, *endVertX*, *startVertY*, *endVertY*, *probA*, *probB*, *probC*, *probD*)

Constructs a selector thread.

See also:
RandomGraphs::PowerLawRandomGraph

7.1.2.2 def ChooseEdges::ChooseEdges::selectVertex (*self*, *sVertX*, *eVertX*, *sVertY*, *eVertY*, *cumulativeA*, *cumulativeB*, *cumulativeC*)

Selects start and end vertices recursively.

Parameters:

sVertX Starting column of the adjacency matrix

eVertX Ending column of the adjacency matrix

sVertY Starting row of the adjacency matrix

eVertY Ending column of the adjacency matrix

cumulativeA Cumulative distribution

cumulativeB Cumulative distribution

cumulativeC Cumulative distribution

Returns:
Selected vertices

7.1.3 Member Data Documentation

7.1.3.1 ChooseEdges::ChooseEdges::endVertX

See also:
RandomGraphs::PowerLawRandomGraph

7.1.3.2 ChooseEdges::ChooseEdges::endVertY

See also:
RandomGraphs::PowerLawRandomGraph

7.1.3.3 ChooseEdges::ChooseEdges::noOfEdges

See also:
RandomGraphs::PowerLawRandomGraph

7.1.3.4 ChooseEdges::ChooseEdges::probA

See also:
RandomGraphs::PowerLawRandomGraph

7.1.3.5 ChooseEdges::ChooseEdges::probB

See also:
RandomGraphs::PowerLawRandomGraph

7.1.3.6 ChooseEdges::ChooseEdges::probC

See also:
RandomGraphs::PowerLawRandomGraph

7.1.3.7 ChooseEdges::ChooseEdges::probD

See also:
RandomGraphs::PowerLawRandomGraph

7.1.3.8 list ChooseEdges::ChooseEdges::serialEdgeList = [ ] [static]

Common serial edge list.

Updated by each thread in a semaphoric operation

7.1.3.9 ChooseEdges::ChooseEdges::startVertX

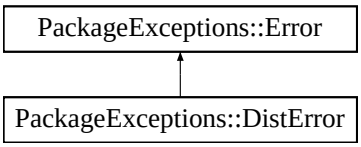
See also:
RandomGraphs::PowerLawRandomGraph

7.1.3.10 ChooseEdges::ChooseEdges::startVertY

See also:
RandomGraphs::PowerLawRandomGraph

7.2 PackageExceptions::DistError Class Reference

Inheritance diagram for PackageExceptions::DistError::



7.2.1 Detailed Description

Represents a DistError exception.

It handles different types of probability distribution related exceptions

Public Member Functions

- def __init__

  Contructs a DistError exception.

Public Attributes

- message

  Error message

7.2.2 Member Function Documentation

7.2.2.1 def PackageExceptions::DistError::__init__ (*self*, *message*)

Contructs a DistError exception.

Parameters:

message Error message

7.3 BaseElements::Edge Class Reference

7.3.1 Detailed Description

Represents graph edge.

Public Member Functions

- def __init__

  Constructs an emtpy edge.

- def __init__

  Constructs a graph edge with given start and end vertices.

- def getStartVertex

  Get the start vertex.

- def getEndVertex

  Get the end vertex.

- def setStartVertex

  Set the start vertex.

- def setEndVertex

  Set the end vertex.

Public Attributes

- startVertex

  Starting vertex of a edge of type BaseElements::Vertex.

- endVertex

  Ending vertex of a edge of type BaseElements::Vertex.

7.3.2 Member Function Documentation

7.3.2.1 def BaseElements::Edge::__init__ (*self*, *startVertex*, *endVertex*)

Constructs a graph edge with given start and end vertices.

Parameters:

startVertex start vertex of the edge

endVertex end vertex of the edge

7.3.2.2 def BaseElements::Edge::getEndVertex (*self*)

Get the end vertex.

Returns:
endVertex End vertex of type BaseElements::Vertex

7.3.2.3 def BaseElements::Edge::getStartVertex (*self*)

Get the start vertex.

Returns:
startVertex Start vertex of type BaseElements::Vertex

7.3.2.4 def BaseElements::Edge::setEndVertex (*self*, *vertex*)

Set the end vertex.

Parameters:
endVertex End vertex of type BaseElements::Vertex

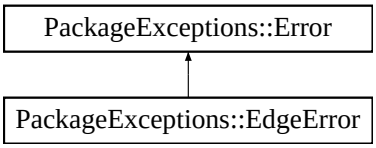
7.3.2.5 def BaseElements::Edge::setStartVertex (*self*, *vertex*)

Set the start vertex.

Parameters:
startVertex Start vertex of type BaseElements::Vertex

7.4 PackageExceptions::EdgeError Class Reference

Inheritance diagram for PackageExceptions::EdgeError::



7.4.1 Detailed Description

Represents a EdgeError exception.

It handles different types of graph edge related exceptions

Public Member Functions

- def __init__

  Contructs a EdgeError exception.

Public Attributes

- edgeNumber

  Vertex number for which the exception occured.

- message

  Error message

7.4.2 Member Function Documentation

7.4.2.1 def PackageExceptions::EdgeError::__init__ (*self*, *edgeNumber*, *message*)

Contructs a EdgeError exception.

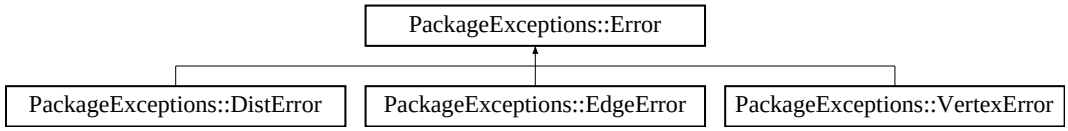
Parameters:

edgeNumber Edge number for which the exception occured

message Error message

7.5 PackageExceptions::Error Class Reference

Inheritance diagram for PackageExceptions::Error::



7.5.1 Detailed Description

Empty base class from which all exceptions are derived.

7.6 PackageExceptions::ErrorMessages Class Reference

7.6.1 Detailed Description

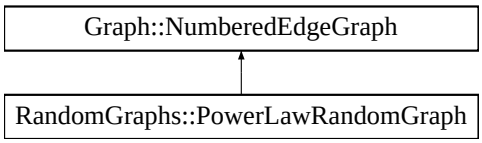
Collection of various error message strings.

Static Public Attributes

- string **vertexAlreadyExists** = 'Vertex number already exists'
- string **vertexNotFound** = 'Vertex number not found'
- string **edgeAlreadyExists** = 'Edge number already exists'
- string **edgeNotFound** = 'Edge number not found'
- string **distAddOne** = 'Probabilities do not add to one'

7.7 Graph::NumberedEdgeGraph Class Reference

Inheritance diagram for Graph::NumberedEdgeGraph::



7.7.1 Detailed Description

Represents a numbered edge graph.

Numbered edges are required to distinguish multiple edges between same set of vertices. This class also provides an indexed vertex and edge sets. These indices have certain advantages while computing in-degree and out-degree distributions.

Public Member Functions

- def __init__
  Constructs a numbered edge graph.
- def addEdge
  Adds an edge to a graph.
- def deleteEdge
  Delete an edge.
- def addVertex
  Adds a vertex.
- def deleteVertex
  Deletes a vertex.
- def getEdges
  Get all graph edges.
- def getVertices
  Get all graph vertices.
- def getOutNeighbors
  Get out-neighbors for a vertex.
- def getInNeighbors
  Get in-neighbors for a vertex.
- def getNumberOfOutNeighbors
  Get number of out-neighbors for a vertex.

- def getNumberOfInNeighbors
  Get number of in-neighbors for a vertex.
- def getInDegreeDistribution
  Get in-degree distribution.
- def getOutDegreeDistribution
  Get out-degree distribution.
- def getVerticesByInDegree
  Gets all the vertices with a particular in-degree.
- def getVerticesByOutDegree
  Gets all the vertices with a particular out-degree.
- def writeEdges
  Write edges to file.
- def readEdges
  Read edges from file.
- def findEdge
  Find edge with a given edge number.
- def findVertex
  Find vertex with a given vertex number.
- def hasVertex
  Checks if vertex is present.

Public Attributes

- edgeIndex
  Dictionary of edges, indexed by edge number.
- vertexIndex
  Dictionary of vertices, indexed by vertex number.

7.7.2 Member Function Documentation

7.7.2.1 def Graph::NumberedEdgeGraph::addEdge (*self*, *edge*)

Adds an edge to a graph.

It also updates the vertex and edge indices.

Parameters:

edge Edge of type BaseElements::Edge to be added to the graph

7.7.2.2 def Graph::NumberedEdgeGraph::addVertex (*self*, *vertexNumber*)

Adds a vertex.

Should be used with care

Parameters:
vertexNumber Vertex number of vertex to be added

Exceptions:
PackageExceptions::VertexError

7.7.2.3 def Graph::NumberedEdgeGraph::deleteEdge (*self*, *edgeNumber*)

Delete an edge.

Parameters:
edgeNumber Edge number to be deleted

7.7.2.4 def Graph::NumberedEdgeGraph::deleteVertex (*self*, *vertexNumber*)

Deletes a vertex.

Should be used with care

Parameters:
vertexNumber Vertex number to be deleted

7.7.2.5 def Graph::NumberedEdgeGraph::findEdge (*self*, *edgeNumber*)

Find edge with a given edge number.

Parameters:
edgeNumber Edge number to look for

Exceptions:
PackageExceptions::EdgeError

Returns:
Matched edge of type BaseElements::Edge

7.7.2.6 def Graph::NumberedEdgeGraph::findVertex (*self*, *vertexNumber*)

Find vertex with a given vertex number.

Parameters:
vertexNumber Vertex number to look for

Exceptions:
PackageExceptions::VertexError

Returns:
Matched vertex of type BaseElements::Vertex

7.7.2.7 def Graph::NumberedEdgeGraph::getEdges (*self*)

Get all graph edges.

Returns:
edgeIndex Dictionary of edges, indexed by edge number

7.7.2.8 def Graph::NumberedEdgeGraph::getInDegreeDistribution (*self*)

Get in-degree distribution.

Returns:
inDegreeDistribution Dictionary indexed on in-degree. Values are the number of nodes for a in-degree

7.7.2.9 def Graph::NumberedEdgeGraph::getInNeighbors (*self*, *vertexNumber*)

Get in-neighbors for a vertex.

Parameters:
vertexNumber Vertex number for which in-neighbors have to be obtained

Returns:
inNeighbors List of in-neighbors. Each element of type BaseElements::Vertex

7.7.2.10 def Graph::NumberedEdgeGraph::getNumberOfInNeighbors (*self*, *vertexNumber*)

Get number of in-neighbors for a vertex.

Parameters:
vertexNumber Vertex number for which number of in-neighbors have to be obtained

Returns:
Number of in-neighbors

7.7.2.11 def Graph::NumberedEdgeGraph::getNumberOfOutNeighbors (*self*, *vertexNumber*)

Get number of out-neighbors for a vertex.

Parameters:
vertexNumber Vertex number for which number of out-neighbors have to be obtained

Returns:
Number of out-neighbors

7.7.2.12 def Graph::NumberedEdgeGraph::getOutDegreeDistribution (*self*)

Get out-degree distribution.

Returns:
outDegreeDistribution Dictionary indexed on in-degree. Values are the number of nodes for a out-degree

7.7.2.13 def Graph::NumberedEdgeGraph::getOutNeighbors (*self*, *vertexNumber*)

Get out-neighbors for a vertex.

Parameters:
vertexNumber Vertex number for which out-neighbors have to be obtained

Returns:
outNeighbors List of out-neighbors. Each element of type BaseElements::Vertex

7.7.2.14 def Graph::NumberedEdgeGraph::getVertices (*self*)

Get all graph vertices.

Returns:
vertexIndex Dictionary of vertices, indexed by vertex number

7.7.2.15 def Graph::NumberedEdgeGraph::getVerticesByInDegree (*self*, *degree*)

Gets all the vertices with a particular in-degree.

Parameters:
degree In-degree to look for

Returns:
degreeNodes List of vertices. Each element of type BaseElements::Vertex

7.7.2.16 def Graph::NumberedEdgeGraph::getVerticesByOutDegree (*self*, *degree*)

Gets all the vertices with a particular out-degree.

Parameters:
degree Out-degree to look for

Returns:
degreeNodes List of vertices. Each element of type BaseElements::Vertex

7.7.2.17 def Graph::NumberedEdgeGraph::hasVertex (*self*, *vertexNumber*)

Checks if vertex is present.

Parameters:

vertexNumber Vertex number of the vertex to check

Returns:

0 if found. 1 if not found

7.7.2.18 def Graph::NumberedEdgeGraph::readEdges (*self*, *fileName*, *format*)

Read edges from file.

Parameters:

fileName File name to read edges from

format Format of input file. Can take values:

'simple' = simple format

7.7.2.19 def Graph::NumberedEdgeGraph::writeEdges (*self*, *fileName*, *format*)

Write edges to file.

Parameters:

fileName File name to store edges in

format Format of output file. Can take values:

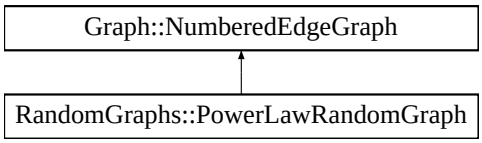
'simple' = simple format

'dot' = format compatible with 'dot' command

Reimplemented in RandomGraphs::PowerLawRandomGraph.

7.8 RandomGraphs::PowerLawRandomGraph Class Reference

Inheritance diagram for RandomGraphs::PowerLawRandomGraph::



7.8.1 Detailed Description

Generates a synthetic Web graph or Power Law graph using an RMAT algorithm.

Public Member Functions

- def __init__

  Constructs an empty graph.

- def setProbs

  Sets the probability with which quadrants in an adjacency matrix are chosen.

- def generate

  Generates a the graph.

- def populate

  Populate graph with edges generated after a call to PowerLawRandomGraph::generate.

- def writeEdges

  Write edges to file.

Public Attributes

- graphSize

  Number of vertices to be considered for generation.

- noOfEdges

  Number of edges to generate.

- probA

  Parameters of the RMAT algorithm. Probability of choosing quadrant A.

- probB

  Probability of choosing quadrant B.

- probC

  Probability of choosing quadrant C.

- probD

  Probability of choosing quadrant D.

- serialEdgeList

  Temporary storage of edges.

- debug

  Debug flag.

- **startVertX**
- **endVertX**
- **startVertY**
- **endVertY**

7.8.2 Member Function Documentation

7.8.2.1 def RandomGraphs::PowerLawRandomGraph::__init__ (*self*, *size*, *noOfEdges*)

Constructs an empty graph.

Parameters:

size Number of vertices to be considered for generation

noOfEdges Number of edges to generate

7.8.2.2 def RandomGraphs::PowerLawRandomGraph::generate (*self*, *noOfThreads*)

Generates a the graph.

Heart of web graph generation algorithm. Each thread gets an equal number of nodes to generate.

Parameters:

noOfThreads Number of threads to spawn for the graph generation. More threads does not correspond to fast generation

7.8.2.3 def RandomGraphs::PowerLawRandomGraph::populate (*self*)

Populate graph with edges generated after a call to PowerLawRandomGraph::generate.

You should call this method before you can use any of the non-overridden method in Graph::Numbered-EdgeGraph

7.8.2.4 def RandomGraphs::PowerLawRandomGraph::setProbs (*self*, *probA*, *probB*, *probC*, *probD*)

Sets the probability with which quadrants in an adjacency matrix are chosen.

Parameters:

probA Probability of choosing quadrant A

probB Probability of choosing quadrant B

probC Probability of choosing quadrant C

probD Probability of choosing quadrant D

Exceptions:

PackageExceptions::DistError

7.8.2.5 def RandomGraphs::PowerLawRandomGraph::writeEdges (*self*, *fileName*, *format*)

Write edges to file.

Parameters:

fileName File name to store edges

format Format of output file. Can take values:

'simple' = simple format

'dot' = format compatible with 'dot' command

Reimplemented from Graph::NumberedEdgeGraph.

7.8.3 Member Data Documentation

7.8.3.1 RandomGraphs::PowerLawRandomGraph::probA

Parameters of the RMAT algorithm. Probability of choosing quadrant A.

Decide the probability with which quadrants in an adjacency matrix are chosen

Todo

Add description about choosing these probabilities

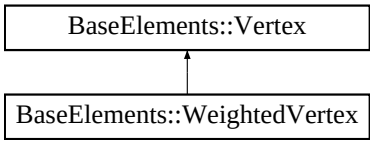
7.8.3.2 RandomGraphs::PowerLawRandomGraph::serialEdgeList

Temporary storage of edges.

Maintained for achieving performance

7.9 BaseElements::Vertex Class Reference

Inheritance diagram for BaseElements::Vertex::



7.9.1 Detailed Description

Represents graph vertex.

Public Member Functions

- def __init__
  Constructs graph vertex given a vertex number.
- def getVertexNumber
  Get vertex number.
- def setVertexNumber
  Set vertex number.

Public Attributes

- vertexNumber
  Vertex number.

7.9.2 Member Function Documentation

7.9.2.1 def BaseElements::Vertex::__init__ (*self*, *vertexNumber*)

Constructs graph vertex given a vertex number.

Parameters:
vertexNumber Vertex number to be assigned to the created vertex

7.9.2.2 def BaseElements::Vertex::getVertexNumber (*self*)

Get vertex number.

Returns:
vertexNumber Vertex number of this vertex

7.9.2.3 def BaseElements::Vertex::setVertexNumber (*self*, *vertexNumber*)

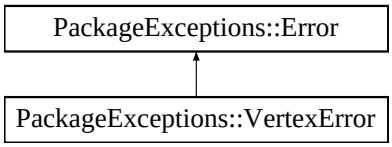
Set vertex number.

Parameters:

vertexNumber New vertex number

7.10 PackageExceptions::VertexError Class Reference

Inheritance diagram for PackageExceptions::VertexError::



7.10.1 Detailed Description

Represents a VertexError exception.

It handles different types of graph vertex related exceptions

Public Member Functions

- def __init__

  Contructs a VertexError exception.

Public Attributes

- vertexNumber

  Vertex number for which the exception occured.

- message

  Error message

7.10.2 Member Function Documentation

7.10.2.1 def PackageExceptions::VertexError::__init__ (*self*, *vertexNumber*, *message*)

Contructs a VertexError exception.

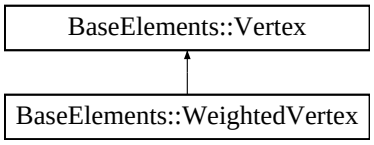
Parameters:

vertexNumber Vertex number for which the exception occured

message Error message

7.11 BaseElements::WeightedVertex Class Reference

Inheritance diagram for BaseElements::WeightedVertex::



7.11.1 Detailed Description

Represents a weighted vertex.

Public Member Functions

- def __init__
  Constructs a weighted vertex given a vertex number and vertex weight.
- def getWeight
  Get vertex weight.
- def setWeight
  Set vertex weight.

Public Attributes

- vertexWeight
  Vertex weight.

7.11.2 Member Function Documentation

7.11.2.1 def BaseElements::WeightedVertex::__init__ (*self*, *vertexNumber*, *vertexWeight*)

Constructs a weighted vertex given a vertex number and vertex weight.

Parameters:

vertexNumber vertex number to be assigned to the created vertex

vertexWeight vertex weight to be assigned to the created vertex

7.11.2.2 def BaseElements::WeightedVertex::getWeight (*self*)

Get vertex weight.

Returns:

vertexWeight Vertex weight of this vertex

7.11.2.3 def BaseElements::WeightedVertex::setWeight (*self*, *vertexWeight*)

Set vertex weight.

Parameters:

vertexWeight New vertex weight

7.12 BaseElements::WeightedVertices Class Reference

7.12.1 Detailed Description

Represents a collection of weighted vertices of type BaseElements::WeightedVertices.

Public Member Functions

- def __init__

  Initialize an empty collection.

- def addVertex

  Add vertex to the collection.

- def delVertex

  Delete vertex from the collection.

- def getVertices

  Get all vertices from the collection.

- def findVertex

  Find vertex in the collection.

- def findWeight

  Find weight of a given vertex.

- def hasVertex

  Checks if vertex is present.

Public Attributes

- weightedVertices

  A dictonary of weighted vertices indexed by vertex numbers and values of type BaseElements::WeightedVertex.

7.12.2 Member Function Documentation

7.12.2.1 def BaseElements::WeightedVertices::addVertex (*self*, *weightedVertex*)

Add vertex to the collection.

Parameters:

weightedVertex Weighted vertex to be added. Should be of type BaseElements::WeightedVertex

7.12.2.2 def BaseElements::WeightedVertices::delVertex (*self*, *vertexNumber*)

Delete vertex from the collection.

Parameters:
vertexNumber Vertex number of the vertex to be deleted

7.12.2.3 def BaseElements::WeightedVertices::findVertex (*self*, *vertexNumber*)

Find vertex in the collection.

Parameters:
vertexNumber Vertex number to be found

Exceptions:
PackageExceptions::VertexError

Returns:
weightedVertex Found weighted vertex of type BaseElements::WeightedVertex

7.12.2.4 def BaseElements::WeightedVertices::findWeight (*self*, *vertexNumber*)

Find weight of a given vertex.

Parameters:
vertexNumber Vertex number of the vertex whose weight is to be found

Returns:
vertexWeight Weight of vertex

7.12.2.5 def BaseElements::WeightedVertices::getVertices (*self*)

Get all vertices from the collection.

Returns:
weightedVertices A dict of all weighted vertices indexed by vertex number

7.12.2.6 def BaseElements::WeightedVertices::hasVertex (*self*, *vertexNumber*)

Checks if vertex is present.

Parameters:
vertexNumber Vertex number to be checked

Returns:
0 if vertex is found. Otherwise 1

Chapter 8

Python Webgraph Generator Page Documentation

8.1 Todo List

Member RandomGraphs::PowerLawRandomGraph::probA Add description about choosing these probabilities

Index