ExpositoTOP

Generated by Doxygen 1.15.0

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

Class Documentation

3.1 es.ull.esit.utilities.BellmanFord Class Reference

Implements the Bellman-Ford algorithm for shortest path calculation.

Public Member Functions

- BellmanFord (int[ ][ ] distanceMatrix, int nodes, ArrayList< Integer > path)
- int[ ] getDistances ()
- int getValue ()
- void **solve** ()

3.1.1 Detailed Description

Implements the Bellman-Ford algorithm for shortest path calculation.

This class provides functionality to compute shortest paths in a weighted directed graph using the Bellman-Ford algorithm. It manages the distance matrix, node and edge lists, and the resulting path and distance values. The class supports initialization with a distance matrix and node count, and exposes methods to solve the shortest path problem, retrieve computed distances, and access the optimal path and its value.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 BellmanFord()

```
es.ull.esit.utilities.BellmanFord.BellmanFord (
             int distanceMatrix[ ][ ],
             int nodes,
             ArrayList< Integer > path)  [inline]
```

Parameters

distanceMatrix	
nodes	
path	

3.1.3 Member Function Documentation

3.1.3.1 getDistances()

```
int[ ] es.ull.esit.utilities.BellmanFord.getDistances ()  [inline]
```

Returns

3.1.3.2 getValue()

```
int es.ull.esit.utilities.BellmanFord.getValue ()  [inline]
```

Returns

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

- src/main/java/es/ull/esit/utilities/BellmanFord.java

3.2 es.ull.esit.utilities.ExpositoUtilities Class Reference

Utility class providing helper methods for formatting, file operations, and matrix manipulation.

Static Public Member Functions

- static int **getFirstAppearance** (int[ ] vector, int element)
- static void **printFile** (String file)
- static String **simplifyString** (String string)
- static double[ ][ ] **multiplyMatrices** (double a[ ][ ], double b[ ][ ])
- static void **writeTextToFile** (String file, String text) throws IOException
- static String **getFormat** (String string)
- static String **getFormat** (double value)
- static String **getFormat** (double value, int zeros)
- static String **getFormat** (String string, int width)
- static String **getFormat** (String string, int width, int alignment)
- static String **getFormat** (ArrayList< String > strings, int width)
- static String **getFormat** (ArrayList< Integer > strings)
- static String **getFormat** (String[ ] strings, int width)
- static String **getFormat** (String[ ][ ] matrixStrings, int width)
- static String **getFormat** (String[ ] strings)
- static String **getFormat** (String[ ] strings, int[ ] width)
- static String **getFormat** (String[ ] strings, int[ ] width, int[ ] alignment)
- static boolean **isInteger** (String str)
- static boolean **isDouble** (String str)
- static boolean **isAcyclic** (int[ ][ ] distanceMatrix)
- static boolean **thereIsPath** (int[ ][ ] distanceMatrix, int node)

Static Public Attributes

- static final int **DEFAULT_COLUMN_WIDTH** = 10
- static final int **ALIGNMENT_LEFT** = 1
- static final int **ALIGNMENT_RIGHT** = 2

3.2.1 Detailed Description

Utility class providing helper methods for formatting, file operations, and matrix manipulation.

This class offers a collection of static utility methods for string formatting, reading and writing files, matrix operations, type checking, and graph-related functions. It supports tasks such as formatting numbers and strings, multiplying matrices, checking for cycles in graphs, and simplifying string content. The utilities are designed to facilitate common operations required in data processing and algorithm implementation.

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

- src/main/java/es/ull/esit/utilities/ExpositoUtilities.java

3.3 top.mainTOPTW Class Reference

Static Public Member Functions

- static void **main** (String[ ] args)

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

- src/main/java/top/mainTOPTW.java

3.4 es.ull.esit.utils.Pair< F, S > Class Template Reference

Generic utility class for storing a pair of objects.

Public Member Functions

- **Pair** (F first, S second)
- boolean **equals** (Object o)
- int **hashCode** ()

Static Public Member Functions

- static< A, B > Pair< A, B > **create** (A a, B b)

Public Attributes

- final F **first**
- final S **second**

3.4.1 Detailed Description

Generic utility class for storing a pair of objects.

This class represents a tuple containing two elements, referred to as 'first' and 'second'. It provides methods for equality comparison, hash code generation, and static creation of pairs. The class is useful for grouping related objects and is commonly used in data structures and algorithms.

Template Parameters

F	Type of the first element.
S	Type of the second element.

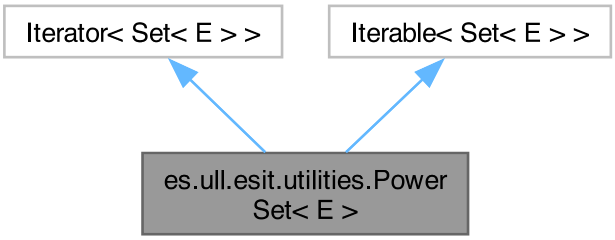
The documentation for this class was generated from the following file:

- src/main/java/es/ull/esit/utils/Pair.java

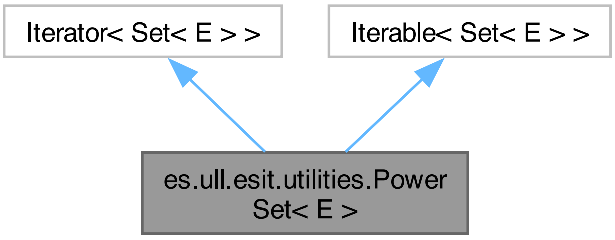
3.5 es.ull.esit.utilities.PowerSet< E > Class Template Reference

Utility class for generating all subsets (power set) of a given set.

Inheritance diagram for es.ull.esit.utilities.PowerSet< E >:



Collaboration diagram for es.ull.esit.utilities.PowerSet< E >:



Public Member Functions

- **PowerSet** (Set< E > set)
- boolean **hasNext** ()
- Set< E > **next** ()
- void **remove** ()
- Iterator< Set< E > > **iterator** ()

3.5.1 Detailed Description

Utility class for generating all subsets (power set) of a given set.

This generic class implements both Iterator and Iterable interfaces to enumerate all possible subsets of a provided set. It uses a BitSet to efficiently track subset membership and supports iteration over the power set. Useful for combinatorial algorithms and exhaustive search tasks.

Template Parameters

E	Type of elements in the set.

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

- src/main/java/es/ull/esit/utilities/PowerSet.java

3.6 top.TOPTW Class Reference

Represents a Team Orienteering Problem with Time Windows (TOPTW) instance.

Public Member Functions

- **TOPTW** (int nodes, int routes)
- boolean **isDepot** (int a)
- double **getDistance** (int[ ] route)
- double **getDistance** (ArrayList< Integer > route)
- double **getDistance** (ArrayList< Integer >[ ] routes)
- void **calculateDistanceMatrix** ()
- double **getMaxTimePerRoute** ()
- void **setMaxTimePerRoute** (double maxTimePerRoute)
- double **getMaxRoutes** ()
- void **setMaxRoutes** (double maxRoutes)
- int **getPOIs** ()
- double **getDistance** (int i, int j)
- double **getTime** (int i, int j)
- int **getNodes** ()
- void **setNodes** (int nodes)
- double **getX** (int index)
- void **setX** (int index, double x)
- double **getY** (int index)
- void **setY** (int index, double y)
- double **getScore** (int index)
- double[ ] **getScore** ()
- void **setScore** (int index, double score)
- double **getReadyTime** (int index)
- void **setReadyTime** (int index, double readyTime)
- double **getDueTime** (int index)
- void **setDueTime** (int index, double dueTime)
- double **getServiceTime** (int index)
- void **setServiceTime** (int index, double serviceTime)
- int **getVehicles** ()
- String **toString** ()
- int **addNode** ()
- int **addNodeDepot** ()

3.6.1 Detailed Description

Represents a Team Orienteering Problem with Time Windows (TOPTW) instance.

This class models the structure and data of a TOPTW problem, including the number of routes, points of interest (POIs), their coordinates, service times, scores, and time windows. It provides methods to set and retrieve these attributes, as well as to calculate the distance matrix required for route optimization algorithms.

The class serves as the main data container for problem instances and supports initialization and manipulation of all relevant parameters for solving the TOPTW.

This class encapsulates all relevant data and operations for a TOPTW problem, including the number of nodes, routes, coordinates, scores, service times, time windows, and the distance matrix. It provides methods to access and modify these attributes, calculate distances between nodes, and manage depots and vehicles. The class serves as the main data container for problem instances and supports initialization and manipulation of all parameters required for solving and optimizing TOPTW routes.

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

- src/main/java/top/TOPTW.java

3.7 top.TOPTWEvaluator Class Reference

Provides evaluation methods for TOPTW solutions.

Public Member Functions

- void **evaluate** (TOPTWSolution solution)

Static Public Attributes

- static double **NO_EVALUATED** = -1.0

3.7.1 Detailed Description

Provides evaluation methods for TOPTW solutions.

This utility class contains methods to assess the quality and objective function value of solutions to the Team Orienteering Problem with Time Windows (TOPTW). It can be extended to implement various evaluation strategies, such as calculating total distance, score, or feasibility of routes. The class is intended to support solution analysis and optimization processes.

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

- src/main/java/top/TOPTWEvaluator.java

3.8 top.TOPTWGRASP Class Reference

Implements the GRASP metaheuristic for solving TOPTW instances.

Public Member Functions

- **TOPTWGRASP** (TOPTWSolution sol)
- void **GRASP** (int maxIterations, int maxSizeRCL)
- int **aleatorySelectionRCL** (int maxTRCL)
- int **fuzzySelectionBestFDRCL** (ArrayList< double[ ] > rcl)
- int **fuzzySelectionAlphaCutRCL** (ArrayList< double[ ] > rcl, double alpha)
- void **computeGreedySolution** (int maxSizeRCL)
- void **updateSolution** (double[ ] candidateSelected, ArrayList< ArrayList< Double > > departureTimes)
- ArrayList< double[ ]> **comprehensiveEvaluation** (ArrayList< Integer > customers, ArrayList< ArrayList< Double > > departureTimes)
- TOPTWSolution **getSolution** ()
- void **setSolution** (TOPTWSolution solution)
- int **getSolutionTime** ()
- void **setSolutionTime** (int solutionTime)
- double **getMaxScore** ()

Static Public Attributes

- static double **NO_EVALUATED** = -1.0

3.8.1 Detailed Description

Implements the GRASP metaheuristic for solving TOPTW instances.

This class provides methods for constructing and improving solutions to the Team Orienteering Problem with Time Windows (TOPTW) using the Greedy Randomized Adaptive Search Procedure (GRASP). It includes routines for greedy randomized construction, candidate selection (including fuzzy logic and alpha-cut strategies), and solution evaluation. The class supports iterative optimization, restricted candidate list management, and integration with solution and problem data.

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

- src/main/java/top/TOPTWGRASP.java

3.9 top.TOPTWReader Class Reference

Utility class for reading and parsing TOPTW problem instances from files.

Static Public Member Functions

- static TOPTW readProblem (String filePath)

  Reads a TOPTW problem instance from a file.

3.9.1 Detailed Description

Utility class for reading and parsing TOPTW problem instances from files.

This class provides static methods to load Team Orienteering Problem with Time Windows (TOPTW) instances from a specified file path. It parses the input file, initializes the problem data, and computes the distance matrix required for further processing.

3.9.2 Member Function Documentation

3.9.2.1 readProblem()

```
TOPTW top.TOPTWReader.readProblem (
            String filePath)  [inline], [static]
```

Reads a TOPTW problem instance from a file.

Parameters

filePath	Path to the input file containing the problem definition.

Returns

A TOPTW object initialized with the parsed data.

This method opens the specified file, reads the problem parameters and points of interest (POIs), sets their coordinates, service times, scores, and time windows, and calculates the distance matrix. If an error occurs during reading, the method prints the error and terminates the program.

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

- src/main/java/top/TOPTWReader.java

3.10 top.TOPTWRoute Class Reference

Represents a route segment in a TOPTW solution.

Public Member Functions

- int **getPredeccesor** ()
- int **getSuccesor** ()
- int **getId** ()
- void **setPredeccesor** (int pre)
- void **setSuccesor** (int suc)
- void **setId** (int id)

3.10.1 Detailed Description

Represents a route segment in a TOPTW solution.

This class models a route segment by storing the predecessor node, successor node, and route identifier. It provides getter and setter methods for these attributes, supporting the construction and manipulation of routes within Team Orienteering Problem with Time Windows (TOPTW) solutions.

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

- src/main/java/top/TOPTWRoute.java

3.11 top.TOPTWSolution Class Reference

Represents a solution to a TOPTW instance.

Public Member Functions

- **TOPTWSolution** (TOPTW problem)
- void **initSolution** ()
- boolean **isDepot** (int c)
- boolean **equals** (TOPTWSolution otherSolution)
- int **getAvailableVehicles** ()
- int **getCreatedRoutes** ()
- double **getDistance** (int x, int y)
- void **setAvailableVehicles** (int availableVehicles)
- int **getPredecessor** (int customer)
- int[ ] **getPredecessors** ()
- TOPTW **getProblem** ()
- double **getObjectiveFunctionValue** ()
- int **getPositionInRoute** (int customer)
- int **getSuccessor** (int customer)
- int[ ] **getSuccessors** ()
- int **getIndexRoute** (int index)
- double **getWaitingTime** (int customer)
- void **setObjectiveFunctionValue** (double objectiveFunctionValue)
- void **setPositionInRoute** (int customer, int position)
- void **setPredecessor** (int customer, int predecessor)
- void **setSuccessor** (int customer, int succesor)
- void **setWaitingTime** (int customer, int waitingTime)
- String **getInfoSolution** ()
- double **evaluateFitness** ()
- int **addRoute** ()
- double **printSolution** ()

Static Public Attributes

- static final int **NO_INITIALIZED** = -1

3.11.1 Detailed Description

Represents a solution to a TOPTW instance.

This class encapsulates the solution structure for the Team Orienteering Problem with Time Windows (TOPTW), including route assignments, predecessor and successor relationships, waiting times, and objective function value. It provides methods for initializing, evaluating, and manipulating solutions, supporting fitness calculation, route management, and solution comparison.

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

- src/main/java/top/TOPTWSolution.java

Index