SB-VRP input/output file(s) specification

Two file formats are provided: a **verbose format**, for non-Euclidian instances and a **compact format**, for instances with Euclidian distances. The reason for keeping two formats is simple: when Euclidian distances are employed, there is no need to report each individual instance. This results in considerably smaller input files.

INSTANCE CONVERTER

Despite the possibility of very large input files, a converter was developed that allows converting an Euclidian instance to the format dedicated to the non-Euclidian ones. The usage is straightforward:

- java -jar sbvrp-tool.jar –c instance.vrp

This converts the file instance.txt into the more verbose non-Euclidian format.

VERBOSE INPUT FORMAT

The verbose format is the one specified by the First VeRoLog Challenge organizers. It consists of several csv/text files. In these files the first line acts as a header and provides the description of what is to come. The data is provided in the remaining lines, separated by semicolons. In total, five files are given: Locations.txt, Vehicles.txt, SwapActions.txt, DistanceTimeCoordinates.txt and DistanceTimesData.txt. Note that all files must be in the same folder – the folder will be the instance name.

Locations.txt

Depots, Swap Locations and Customers data are given in this file, where for each location is reported:

- location type (DEPOT, SWAP_LOCATION or CUSTOMER)
- descriptive name, the post code and the city (for descriptive purposes only)
- X and Y coordinates of the geographic point of the location: the pair of X,Y coordinates of the location will be used as identifier to determine the distance and time of the trip to other locations in the distance and time tables.
- quantity to be delivered in load units, for customers only
- possibility of being visited by a train (1=possible, 0=impossible), for customers only
- possibility of performing a swap (1=yes, 0=no), for swap locations only

Vehicles.txt

For each equipment type (TRUCK, SEMI_TRAILER and SWAP_BODY) is reported:

- type name
- capacity in load units
- distance cost in monetary units per Km
- time cost in monetary units per hour
- fixed cost in monetary units per use

SwapActions.txt

For swap action type (PARK, SWAP, EXCHANGE and PICKUP) is reported:

- type name;
- fixed duration in minutes.

DistanceTimesCoordinates.txt

This file contains a list of X and Y coordinates of the geographic points corresponding to locations.

DistanceTimesData.txt

This file presents, for each point corresponding to location (DEPOT, SWAP_LOCATION and CUSTOMER) in the same order as reported in the file DistanceTimesCoordinates.txt:

- the distance d_{ij} and driving period p_{ij} between the start-location i and each destination-location j are stored in separate rows separated by a semicolon.

  For example, a distance table that consists of the distances and driving periods between 3 locations is stored as follows:

  $d_{11}; p_{11}; d_{12}; p_{12}; d_{13}; p_{13}$

  $d_{21}; p_{21}; d_{22}; p_{22}; d_{23}; p_{23}$

  $d_{31}; p_{31}; d_{32}; p_{32}; d_{33}; p_{33}$

Important: note that distances in the matrix are given in meters while times are given in seconds.

COMPACT INPUT FORMAT

The compact input format is specified to compactly represent Euclidean instances. Each instance consist of a single file, organized as follows:

Header section with [key] : [value] pairs:

- NAME, COMMENT, TYPE, EDGE_WEIGHT_TYPE: provides additional (optional) information about the instance file
- CUSTOMERS: the number of customers
- SWAP_LOCATIONS: the number of swap locations
- CAPACITY: the capacity of a swap body
- TRUCK_COST: the cost for an edge traveled by a truck without semi-trailer is calculated by multiplying the euclidean distance by this factor
- TRAIN_COST: the cost for an edge traveled by a truck with semi-trailer attached is calculated by multiplying the euclidean distance of the edge by this factor

The locations are informed in the next sections, with one line per location:

- DEPOT_SECTION
  - Location-name x-coord y-coord

- CUSTOMER_SECTION
  - Location-name x-coord y-coord demand train_possible(true/false)

- SWAP_SECTION
  - Location-name x-coord y-coord

OUTPUT FORMAT

A single output format is considered, consisting of a csv file. Each line of the file represents one stop of a vehicle. The csv file has 10 columns

- TOUR_ID: ID of the tour (or vehicle), stating with an R (example: R01, R02, etc.);
- TOUR_POSITION: position of the current location in the tour (starting at 1);
- LOCATION_ID: ID of the location indicated in the line;
- LOCATION_TYPE: type of location (DEPOT, CUSTOMER or SWAP_LOCATION)
- SEMI_TRAILER_ATTACHED: indicates if the semi-trailer is attached at this stop;
- SWAP_BODY_TRUCK: indicates if the first swap body is attached at this stop;
- SWAP_BODY_SEMI_TRAILER: indicates if the second swap body is attached at this stop;
- SWAP_ACTION: if the location is a swap location, this indicates the action; possible values are: NONE, PARK, PICKUP, SWAP and EXCHANGE
- SWAP_BODY_1_QUANTITY: amount consumed at this stop from the first swap body;
- SWAP_BODY_2_QUANTITIY: amount consumed at this stop from the second swap body.

Many examples of both instances and solutions can be downloaded from our website:
http://benchmark.gent.cs.kuleuven.be/sbvrp