Inventory routing and on-line inventory routing file format

Marc Sevaux, Martin J. Geiger

Inventory Routing and On-line Inventory Routing File Format

M. Sevaux[1,2] M. J. Geiger[1]

[1]Helmut-Schmidt-Universität, Logistics Management Department, Hamburg, Germany
[2]Université de Bretagne-Sud, Lab-STICC, Lorient, France

January 2011

Abstract

This document presents a simple extension of the TSPLIB file format to serve our needs in the Inventory Routing Problem types. Instead of creating a new file format or putting ASCII files online with a simple description, we have chosen to extend the TSPLIB file format.

This document is an extension of the TSPLIB file format description proposed in [1] to be used for the Inventory Routing Problem and variants. No repetition of the information in [1] is given, unless strictly necessary, but the present file is self-contained to describe our instances.

TSPLIB is a library of sample instances for the TSP (and related problems) from various sources and of various types. In addition to the following problem classes TSP, HCP, ATSP, SOP and CVRP described in [1], we have created two new classes called:

Inventory routing problem (IRP)

We are given $n-1$ nodes, one depot and distances from the nodes to the depot, as well as between nodes. For delivery to the nodes, trucks with identical capacities are available. The problem is defined for a known a finite horizon $1 \dots H$. For each time period, each node has a demand. Each node has also a maximum inventory level. An initial stock is also associated with the node. Potentially, for the depot, an initial stock and a production for all periods might be given. The problem is to find tours for the trucks that satisfy the node demands over the whole horizon without violating the truck capacity constraint and the node maximum inventory level constraint. The number of trucks is not specified. Each tour visits a subset of the nodes and starts and terminates at the depot.

On-line inventory routing problem (OIRP)

The problem definition is the same as above except for the demand, but the data is more complex. Instead of having a complete overview of the demand for the whole horizon, at the beginning of each time period, and for each node, the user will be given a forecast of the forthcoming demand and the precise demand for the current period. This forecast is rather accurate but not precise enough to calculate the exact demand of the next period. The forecast is represented by several figures corresponding to the expected cumulative demand

of a range of consecutive periods starting from the current period. For example, 3 figures can be given, representing the expected cumulative demand of the next 5 periods, the expected cumulative demand of the next 20 periods, and the expected cumulative demand of the next 60 periods.

1 The file format

Each file consists of a **specification part** and a **data part**. The specification part contains information on the file format and on its contents. The data part contains explicit data.

1.1 The specification part

All entries in this section are of the form <*keyword*> : <*value*>, where <*keyword*> denotes an alphanumerical keyword and <*value*> denotes alphanumerical or numerical data. The terms <*string*>, <*integer*> and <*real*> denote character string, integer or real data, respectively. The order of specification of the keywords in the data file is arbitrary (in principle), but must be consistent, i. e., whenever a keyword is specified, all necessary information for the correct interpretation of the keyword has to be known. Below we give a list of all available keywords.

1.1.1 `NAME` : <*string*>

Identifies the data file.

1.1.2 `TYPE` : <*string*>

Specifies the type of the data. Possible types are

`IRP` Inventory routing problem data

`OIRP` On-line inventory routing problem data

1.1.3 `COMMENT` : <*string*>

Additional comments (usually the name of the contributor or creator of the problem instance is given here).

1.1.4 `OBJECTIVE` : <*string*>

Since the problem is vast, we would like to allow different people to contribute, and we have let the opportunity here to choose their own objective function.

Possible objective functions are

`ROUTING` Minimise the total routing cost (sum of routing costs/tour lengths for all periods).

`BI_OBJECTIVE` Both the total routing cost and the total inventory cost are taken into account. Inventory cost is the sum of inventory levels at each node at the end of each period.

`SPECIAL` The cost function is described later in the file in the section named `COST_FUNCTION`.

1.1.5 `COST_FUNCTION` : <*string*>

If the `OBJECTIVE` specification is `SPECIAL`, this section should describe the special cost function used (e. g. a linear combination of different objectives).

1.1.6 `DIMENSION` : <*integer*>

For a TSP or ATSP, the dimension is the number of its nodes. For a CVRP, IRP or OIRP, it is the total number of nodes and depots. For a TOUR file it is the dimension of the corresponding problem.

1.1.7 `CAPACITY` : <*integer*>

Specifies the truck capacity in a CVRP, IRP or OIRP.

1.1.8 `PERIODS` : <*integer*>

Specifies the number of periods in an IRP or OIRP

1.1.9 `EDGE_WEIGHT_TYPE` : <*string*>

Specifies how the edge weights (or distances) are given. The values are

`EUC_2D_INT` Weights are Euclidean distances in 2-D (integer rounding)

`EUC_2D_DBL` Weights are Euclidean distances in 2-D (real value)

`EUC_2D_1DD` Weights are Euclidean distances in 2-D (real value but rounded with 1 decimal digits)

Note that the original `EUC_2D` has been replaced by the `EUC_2D_INT`.

Up to now we have used only the `EUC_2D` specification which is the most commonly used for CVRP and IRP problem types.

Some of the sections used in the original paper are not relevant if we limited our usage to the EUC_2D specification. We refer the reader to the original paper [1] for a complete description of other sections.

1.1.10 NODE_COORD_TYPE : <*string*>

Specifies whether coordinates are associated with each node (which, for example may be used for either graphical display or distance computations). The values are

TWOD_COORDS Nodes are specified by coordinates in 2-D

THREED_COORDS Nodes are specified by coordinates in 3-D

NO_COORDS The nodes do not have associated coordinates

Since the default value is NO_COORDS, it is necessary to choose the right option here, especially for the EDGE_WEIGHT_TYPE specification compatibility.

1.1.11 EOF :

Terminates the input data. This entry is optional.

1.2 The data part

Depending on the choice of specifications some additional data may be required. This data is given in corresponding data sections following the specification part. Each data section begins with the corresponding keyword. The length of the section is either implicitly known from the format specification, or the section is terminated by an appropriate end-of-section identifier.

1.2.1 NODE_COORD_SECTION :

Node coordinates are given in this section. Each line is of the form

<*integer*> <*real*> <*real*>

if NODE_COORD_TYPE is TWOD_COORDS. Note that for some instances, coordinates are integer. Think about automatic conversion while reading the file.

The integers give the number of the respective nodes. The real numbers give the associated x and y coordinates.

1.2.2 DEPOT_SECTION :

Contains a list of possible alternate depot nodes. This list is terminated by a -1.

1.2.3 MULTIPLE_PERIOD_DEMAND_SECTION :

The demands of all nodes of an IRP are given in the form (per line)

<integer> <integer> <integer>

The first integer specifies a node number, the second the period identifier, and the third its demand for the specified period. The depot nodes should not occur in this section since their demand is null for all periods.

1.2.4 INVENTORY_LEVEL_SECTION :

The maximum inventory level at all nodes of an IRP are given in the form (per line)

<integer> <integer>

The first integer specifies a node number and the second the maximum inventory level. Please note that this inventory level is a hard constraint at any time. It means that a customer with a maximum inventory level of 100 and 50 units in stock cannot be delivered 60 units in a period, even if its consumption for the period is 20.

1.2.5 INITIAL_INVENTORY_SECTION :

The initial inventory level at all nodes and at the depot of an IRP are given in the form (per line)

<integer> <integer>

The first integer specifies a node number (or the depot identifier), and the second integer the initial inventory level.

1.2.6 DEPOT_SUPPLY_SECTION :

To be compatible with [2], we have added this special section which specifies the number of units that are produced at the depot. The units produced at each period of an IRP are given in the form (per line)

<integer> <integer>

The first integer specifies a period identifier and the second integer the number of units produced. Note that the unit produced in period t cannot be used before period $t + 1$.

1.2.7 FORECAST_DEMAND_SECTION :

If the TYPE specification is OIRP, then this section should provide information on the demand forecast.

The forecast at all nodes and for all periods are given in the form (per line)

<integer> <integer> <integer> <integer>

The first integer specifies a node number, the second integer the period index, the third integer the number of periods in the forecast,and the last integer the expected cumulative demand for the number of periods.

For example, an entry like

3 12 5 150

indicates that for node 3 at period 12 and for the next five periods including the current period (so for periods 12 to 16), the expected cumulative demand is 150. It does not mean that the demand will be 30 in each of the periods 12 to 16!

Note that the forecast might be wrong, but we hope that the people given the forecast are good enough not to deviate too much from reality.

2 The distance functions

For the various choices of `EGDE_WEIGHT_TYPE`, we now describe the computations of the respective distances. In each case we give a (simplified) C-implementation for computing the distances from the input coordinates.

In some cases, distances are required to be integral. In that case, we round to the nearest integer (as rounding is defined mathematically – if $\geq x.5$ rounds up to $x+1$, otherwise rounds down to x). Below we have named the rounding function "`nint`".

2.1 Euclidean distance (L_2-metric)

For edge weight type `EUC_2D_INT`, floating point coordinates must be specified for each node. Let `x[i]` and `y[i]` be the coordinates of node i.

In the 2-dimensional case `EUC_2D_INT`, the distance between two points i and j is computed as follows:

```
xd = x[i] - x[j];
yd = y[i] - y[j];
dij = nint( sqrt( xd*xd + yd*yd) )
```

For edge weight type `EUC_2D_DBL`, the distance between two points i and j is computed as follows:

```
xd = x[i] - x[j];
yd = y[i] - y[j];
dij = sqrt( xd*xd + yd*yd)
```

For edge weight type `EUC_2D_1DD`, the distance between two points i and j is computed as follows:

```
xd = x[i] - x[j];
yd = y[i] - y[j];
dij = 1/10*floor(10*sqrt( xd*xd + yd*yd))
```

This distance measure has been used in [3] but the user should be aware that this function really depends on the coordinate values.

The function `sqrt` is the C square root function and `floor` is the classical mathematical floor function that rounds down real numbers to the integer below.

3 Access

IRP and OIRP instances are available at `logistik.hsu-hh.de/IRP`

References

[1] G. Reinelt. Tsplib95. Internal report, Universität Heidelberg, Heidelberg, Germany, 1995. `http://neo.lcc.uma.es/radi-aeb/WebVRP/data/Doc.ps`.

[2] C. Archetti, L. Bertazzi, A. Hertz, and M-G. Speranza. A hybrid heuristic for an inventory-routing problem. *INFORMS Journal on Computing*, 2011. accepted.

[3] N. Kohl, J. Desrosiers, O.B.G. Madsen, M.M. Solomon, and F. Soumis. 2-path cuts for the vehicle routing problem with time windows. *Transportation Science*, 33(1):101–116, 1999.