jCW User Guide

Win/Vista Deployment
v.2008.09.25

Jorge E. Mendoza (jor-mend@uniandes.edu.co)
Andrés Medaglia (amedagli@uniandes.edu.co)
Nubia Velasco (nvelasco@uniandes.edu.co)
Christelle Guéret (Christelle.Gueret@emn.fr)

Centro para la Optimización y Probabilidad Aplicada (COPA)
Departamento de Ingeniería Industrial
Universidad de los Andes
Colombia
url: http://copa.uniandes.edu.co
e-mail: copa@uniandes.edu.co

Equipe Systèmes Logistiques et de Production (SLP)
Institut de Recherche en Communications et en Cybernétique de Nantes
url: http://www.irccyn.ec-nantes.fr/Slp
France

Created: September 25, 2008
Updated: September 25, 2008

TABLE OF CONTENTS

WHAT IS JCW?

Java Clarke and Wright (jCW) is an object-oriented framework for the rapid development of vehicle routing prototypes based on the savings heuristics (Clarke and Wright, 1964). The framework provides a collection of abstract and concrete Java classes that allow for the implementation of savings-based algorithms that can be used at the core of a standalone application or embedded into a more complex algorithm.

This version of jCW tackles the Distance-constrained Capacitated Vehicle Routing Problem (DCVRP) with the following features:

- there is a single depot (node 0);
- the matrix distance is symmetric;
- the vehicles are capacitated;
- all vehicles are identical (they have the same capacity);
- there is a maximum route length (or time);
- customers may have different service times while served by the vehicle; and
- the number of routes is found by the algorithm (decision variable).

Other variants of the Vehicle Routing Problem (VRP) can be handled by jCW.

LICENSE INFORMATION

The authors allow the use of JCW without charge for research purposes as a member of a non-commercial and academic institution, e.g., a university.

Any publication must include an acknowledgment and a reference to the corresponding publication:

> Mendoza, J. E., Guéret, C., Medaglia, A. L., Velasco, N., and Villegas, J. G. (2008). JCW: an object-oriented framework for the rapid development of vehicle routing heuristics based on savings. XIV Congreso Latino Ibero Americano de Investigación de Operaciones (CLAIO). Cartagena, Colombia. pp. 60-60. ISBN: 978 958 825283-4.

COPA (Universidad de los Andes) and SLP (Ecole des Mines de Nantes) promote the use of industrial applications of optimization and are interested in commercial applications of JCW. Nevertheless, whenever you add value by using JCW, you need a commercial, development, or deployment license.

If you are applying JCW in an industrial setting, you need a commercial license from either Universidad de los Andes or the Ecole des Mines de Nantes. A development license is needed if JCW is used internally in your company. Such a license is needed for: developing nonacademic software which uses JCW, selling results that were obtained by using JCW, and using JCW internally for nonacademic purposes. If you are selling a product that includes (parts of) JCW, you need to obtain one deployment license per copy of the product that you sell. JCW is provided "as is". We do not offer maintenance, but will be glad to hear any problems or suggestions that lead to better versions of JCW. The license agreements are available upon request by contacting the authors at copa@uniandes.edu.co.

REQUIREMENTS

- Windows operating system
- Upon request, it is possible to have a version that runs in any platform in which Java runs (e.g., Windows, Linux, Solaris)

HOW TO RUN JCW?

After installing the setup program, open a command line window and type:

```
jcw <main-configuration-file-path>
```

where <main-configuration-file-path> is the path to the main configuration file.

To test the installation, run the following instance of the Golden et al. (1998) problems that comes with the installation package. Type:

```
jcw ./configfiles/01_D241-10k_configFile.ini
```

CONFIGURATION FILES

jCW has two configuration text files. The main configuration file has the parameters described in Table 1 and illustrated with an example in Figure 1. Note that line comments are possible starting the line with a # character.

Table 1: Fields in the main configuration file

Property Field	Description
NODE	Points to the Java class that describes the node object used in the algorithm. It depends on the type of VRP being solved.
ARC	Points to the Java class that describes the arc object used in the algorithm. It depends on the type of VRP being solved.
ROUTE	Points to the Java class that describes the route object used in the algorithm. It depends on the type of VRP being solved.
NUMBER_OF_CONSTRAINTS	Defines the number of constraints to be handled. For example, for the DCVRP there are two constraints: 1) maximum route length (distance) and 2) vehicle capacity.
CONSTRAINT_*n*	Points to the class that defines the *n*-th constraint, where *n* is an integer number. There should be as many CONSTRAINT_*n* definitions as NUMBER_OF_CONSTRAINTS.
SAVINGS_HEURISTIC	Points to the class that holds the main logic of the algorithm. For example, the savings list could be statically or dynamically calculated depending on the variant of VRP handled.
DATA_MANAGER	This class holds the file readers.
PROBLEM_DATA_FILE	Points to the data configuration file that describes the location of the data sources.
TRACELEVEL	It is an integer that defines the level of verbosity of the output. It is the result of the sum of the following flags: JCW_ERROR = 1 =2^0 JCW_WARNING = 2 =2^1 JCW_TRACE = 4 =2^2 JCW_DEBUG = 8 =2^3 Example: TRACELEVEL=7 turns on ERROR+WARNING+TRACE because it is the sum of 1+2+4.

```
# ========================================================================
# JCW Sample ConfigFile
# ========================================================================
# JCW Configuration File : 01_D241-10k.config.ini
# Instance               :  01_D241-10k - Golden et al. (1998)
# Created                :  2008/09/21
# Created by             :  Andres Medaglia
# Created for            :  DVRP paper (DSS)
# ------------------------------------------------------------------------
NODE                  = edu.uniandes.copa.jcw.DCVRPNode
ARC                   = edu.uniandes.copa.jcw.DCVRPArc
ROUTE                 = edu.uniandes.copa.jcw.DCVRPRoute
NUMBER_OF_CONSTRAINTS= 2
CONSTRAINT_1          = edu.uniandes.copa.jcw.DistanceConstraint
CONSTRAINT_2          = edu.uniandes.copa.jcw.CapacityConstraint
SAVINGS_HEURISTIC     = edu.uniandes.copa.jcw.BasicSavingsHeuristic
DATA_MANAGER          = edu.uniandes.copa.jcw.StandardDataReader
PROBLEM_DATA_FILE     = ./configFiles/01_D241-10k.ini
TRACELEVEL            = 7
# ------------------------------------------------------------------------
```

Figure 1: Example of the main configuration file `01_D241-10k_configFile.ini`

The data configuration file has the parameters described in Table 2. This particular configuration file is tied to the reader class defined in DATA_MANAGER. Examples of the demand, capacity, and distance files are illustrated in Figure 3, Figure 4, and Figure 5, respectively.

Table 2: Fields in the data configuration file

Property Field	Description
DEMAND_FILE	A CSV file (comma-separated values) with information of the node ID, service time, product ID, and demand. For the DCVRP the field product ID is set to 1. This field is used in the multi-compartment VRP.
CAPACITY_FILE	A CSV file (comma-separated values) with the maximum length (distance) and vehicle capacity.
DISTANCE_FILE	A CSV file (comma-separated values) with information of the tail node, head node, and distance.

```
# ========================================================================
# JCW Sample data file
# ========================================================================
# Problem settings File : 01_D241-10k.ini
# Instance              : 01_D241-10k - Golden et al. (1998)
# Created               : 2008/09/21
# Created by            : Andres Medaglia
# Created for           : DVRP paper (DSS)
# ------------------------------------------------------------------------
DEMAND_FILE           = ./dataFiles/01_D241-10k-demandFile.csv
CAPACITY_FILE         = ./dataFiles/01_D241-10k-capacityFile.csv
DISTANCE_FILE         = ./dataFiles/01_D241-10k-distanceFile.csv
# ------------------------------------------------------------------------
```

Figure 2: Example of the data configuration file `01_D241-10k.ini`

```
#-----------------------------------------------------------------------
# Demand file (node data)
#-----------------------------------------------------------------------
# Automatically generated by tsplib2jcw.sas on 21SEP08:14:19:58
# Instance source file: '01_D241-10k.dat'
# Service time precision: 8.0
# Demand precision: 8.0
#-----------------------------------------------------------------------
# Centro de Optimización y Probabilidad Aplicada (COPA).
# e-mail: copa@uniandes.edu.co
# url:    http://copa.uniandes.edu.co
#-----------------------------------------------------------------------
# CSV format: node ID, service time, product ID, demand
# Note: product ID is set to 1 for a single-product VRP.
#-----------------------------------------------------------------------
1,0,1,10
2,0,1,30
3,0,1,30
4,0,1,10
5,0,1,10
6,0,1,30
7,0,1,30
8,0,1,10
9,0,1,10
10,0,1,30
```

Figure 3: Fragment of the demand file `01_D241-10k-demandFile.csv`

```
#-----------------------------------------------------------------------
# Capacity file
#-----------------------------------------------------------------------
# Automatically generated by tsplib2jcw.sas on 21SEP08:14:19:58
# Instance source file: '01_D241-10k.dat'
# Max. capacity precision: 20.0
# Max. length precision: 20.0
#-----------------------------------------------------------------------
# Centro de Optimización y Probabilidad Aplicada (COPA).
# e-mail: copa@uniandes.edu.co
# url:    http://copa.uniandes.edu.co
#-----------------------------------------------------------------------
# CSV format:
# 0, max. length
# 1, max. capacity
#-----------------------------------------------------------------------
0,650
1,550
```

Figure 4: Example of the capacity file `01_D241-10k-capacityFile.csv`

```
#-----------------------------------------------------------------------
# Distance file
#-----------------------------------------------------------------------
# Automatically generated by tsplib2jcw.sas on 21SEP08:14:19:57
# Instance source file: '01_D241-10k.dat'
# Distance precision: 20.2
#-----------------------------------------------------------------------
# Centro de Optimización y Probabilidad Aplicada (COPA).
# e-mail: copa@uniandes.edu.co
# url:    http://copa.uniandes.edu.co
#-----------------------------------------------------------------------
# CSV format: tail-node-ID, head-node-ID , distance
#-----------------------------------------------------------------------
1,2,4.71
1,3,9.39
1,4,14.01
1,5,18.54
1,6,22.96
1,7,27.24
1,8,31.35
1,9,35.27
1,10,38.97
1,11,42.43
1,12,45.62
1,13,48.54
1,14,51.16
1,15,53.46
```

Figure 5: Fragment of the distance file `01_D241-10k-distanceFile.csv`

HISTORY OF CHANGES

- Version v.2004.11.15
  - First running version.
  - Several of the CMT instances from Christofides, Mingozzi & Toth (1979) were successfuly run.
  - An example from Larson & Odoni (1981) was tested with C&W.
- Version v.2006.07.20
  - First trials with EAAB data by Jorge Mendoza.
- Version v.2006.10.16
  - Compiled to an executable with Excelsior JET 3.70.
  - Routes are output to file with CSV format.
  - The file name is given in the configuration file parameter OUTPUT_FILE.
  - The fields in this file are:
    - RouteID: internal number related to a node.
    - NodeID: node, related to column JDBC_NODES_COL_ID.
    - X: X-coordinate, related to column JDBC_NODES_COL_LONGITUDE.
    - Y: Y-coordinate, related to column JDBC_NODES_COL_LATITUDE.
    - Order: sequence in route V. Starts with -1 and ends in |V|, both of these nodes refer to the depot.
    - Load: "demand" related to column JDBC_NODES_COL_DEMAND
- Version v.2007.02.14
  - Trial by Mapas y Datos using data from Codensa.
  - First trials with MS Access as database.
  - A problem was fixed on JDBCWriter.createArcsDistanceTable().

```
//FIX [20070214]: Index out of bounds
//String tableName = arcsOutputTableName.substring(0,arcsOutputTableName.indexOf("$"));
String tableName = arcsOutputTableName;
```

  - Solved real world instance with 628 nodes with data source located at: `C:\Andres\Uniandes\Consultoria\Propuestas\MyD-Codensa\Codensa.mdb`
  - Sample configuration file (note the absence of $ in the table names):

- Version v.2008.08.28
  - jCW has been redesigned to manage the concept of dynamic constraint handling. Constraints are now dynamically loaded and checked at run time while merging routes.
  - Even though the concept is general enough, it remains hard coded to the DCVRP.
  - Some extensions for the multi-compartment VRP are also present in the current design.
  - A C&W dynamic variant has been implemented to update the savings list after mergers.
  - The database readers via JDBC have been replaced for CSV readers, although the former JDBC readers could be refurbished to meet the contract of the new data manager class.

- Version v.2008.09.25
  - Compiled with Excelsior JET 5.0.
  - Added license agreement.

TO DO LIST

- 2006/10/16
  - Implement 2-opt and 3-opt post optimization procedures.
  - Output route to data source via JDBC (new table).
- 2008/09/26
  - Recover JDBC readers.
  - Revisit code so that it becomes untangled to the DCVRP instance.

CONTACT INFORMATION

For further technical information or license queries please contact: copa@uniandes.edu.co

REFERENCES

N. Christofides, A. Mingozzi, and P. Toth (1979). The vehicle routing problem. In N. Christofides, A. Mingozzi, P. Toth, and C. Sandi, eds. Combinatorial Optimization, 315-338. Wiley.

G. Clarke and J. V. Wright (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12(4):568-581.

B. Golden, E. Wasil, J. Kelly, and I. Chao. (1998). The impact of metaheuristics on solving the vehicle routing problem: algorithms, problem sets, and computational results. In T. Crainic and G. Laporte, eds. Fleet management and logistics, 33-56. Kluwer.

R. C. Larson and A. Odoni (1981). Urban Operations Research. New Jersey: Prentice-Hall.

Mendoza, J. E., Guéret, C., Medaglia, A. L., Velasco, N., and Villegas, J. G. (2008). JCW: an object-oriented framework for the rapid development of vehicle routing heuristics based on savings. XIV Congreso Latino Ibero Americano de Investigación de Operaciones (CLAIO). Cartagena, Colombia. pp. 60-60. ISBN: 978 958 825283-4.