A CONSTRAINT PROGRAMMING-BASED LIBRARY FOR THE VEHICLE ROUTING PROBLEM

Daniel Riera[a], Àngel A. Juan[b], Daniel Guimarans[c] Estel·la Pagans[d]

[a][b]GRES-UOC/DPCS. EIMT. Universitat Oberta de Catalunya
[c] Telecommunication and System Engineering Dept., Universitat Autònoma de Barcelona, Spain
[d]Odournet S.L., Barcelona, Spain

[a]drierat@uoc.edu, [b]ajuanp@uoc.edu, [c] daniel.guimarans@uab.cat, [d] epagans@uoc.edu

ABSTRACT
This paper presents a first approach to a constraint programming-based library for the modelling of general vehicle routing problems (VRP). In this work we present a set of constraints and cost functions written in ECLiPSe, a well known Constraint Programming (CP) language, which can be combined to build different VRP models without rewriting.

Keywords: vehicle routing problem, constraint programming, library

1. INTRODUCTION

Libraries are a nice facility when a researcher is trying to solve a problem and wants to improve something somebody else has proposed before. Although previous results are usually published and easily accessible, often it is not easy to reach these results the same way the original author did. This means a lot of time and perhaps a few e-mails to be able to exactly repeat something already existing.

Our research groups (i.e. GRES-UOC and DPCS) work on hybrid approaches to solve NP problems. Currently, the chosen one is the VRP (Golden, Raghavan, and Wasil, 2008). The VRP is a well known NP-Hard problem consisting in the service of a set of clients - spread in a map - by a set of vehicles. This is the generalisation of multiple different specific problems, which differ in the definition of the constraints applied to the basis. Thus, a few examples are: the CVRP, which contain constraints on the capacity of the vehicles; the VRPTW which includes time windows when clients can be served; etc.

Although there are specific approaches for these complex problems, single techniques often fall short to tackle real world instances. These can be easily seen because results are available only for laboratory problems, or because models become a simplification of the real problem. The later usually makes too big the distance between the solution and the original problem, and thus it becomes useless.

A common solution to the problems presented in the previous paragraph is the hybridization. It implies the use of different techniques/methodologies like artificial intelligence, operations research or constraint programming in order to cooperate and find better ways to reach a solution. These techniques have their advantages and disadvantages, but it is very important for us to be able to build a model in any of their formalisms as quick as possible. A common mistake is the adaptation of models built for a specific methodology - like simplex, for instance - to another one - like constraint programming. Since the solving algorithms, methods, etc. are different, the model written thinking in one of these technologies is terribly slow and inefficient when used in the other.

Researchers are usually experts in several methodologies but they do not need to be in all of them. The use of a library thought for a specific programming methodology - in this case, constraint programming - helps them and allows researchers to think about the problem itself and avoid the modelling issues.

This paper presents a first approach to develop a library for modelling the VRP by means of CP. The main aim of the paper is to offer a tool not only for being used by our research groups but also by the research community working around the VRP. The use of this library leaves the constraint programming model implementation hidden from the user. Thus, programming issues are leaved apart and model construction becomes straightforward.

The structure of the paper is as follows. After this introduction, section 2 talks about the problem studied (the vehicle routing problem) and the different instances which can be found in literature. Section 3 gives an overview of the techniques commonly used to tackle the VRP. Later, sections 4 and 5 present the mathematical model - as a set of variables and constraints on those variables - used for the library construction and the library Prolog/ECLiPSe (Sterling and Shapiro, 1986) predicates corresponding to those constraints. While section 6 explains how to create a file with a VRP

model, section 7 shows the results of a case study. Finally, the conclusions and future work conclude the paper in section 8.

2. VEHICLE ROUTING PROBLEMS

The Vehicle Routing Problem (VRP), as told before, is an NP-hard problem in which a set of customers' demands have to be served by a fleet of vehicles departing from a certain number of depots. This is the basis to a very heterogeneous family of problems which can include extra constraints, enriching the possible model of the problem to solve. Furthermore, there are some costs associated with the resolution of these problems. In particular, it is usual to explicitly consider costs due to moving a vehicle from one node - customer or depot - to another. The classical goal here consists on determining the optimal set of routes that minimizes those tangible costs under a certain set of predefined constraints.

Depending on the considered constraints there are a number of VRP versions, but firstly, there is an initial classification to consider: On the one hand, there are the static VRPs. These are all those problems with no additional elements to be incorporated in real-time. Thus, all the information is known from the beginning. They are clearly static and deterministic. On the other hand there are problems which consider that customers, constraints or others might be added later, and hence, a re-routing would be needed on-line. These are called dynamic VRPs. Although usually, the main difference between these two versions should not have too much to do with the problem model and the optimisation strategy, when working with stochastic demands the model might need some adaptation. In the current work stage, dynamic VRPs with stochastic demands are not considered but everything said further can be applied to the rest of static and dynamic VRPs.

These are some of the most studied VRPs:

- CVRP (Capacitated VRP): Vehicles have limited carrying capacities
- VRPTW (VRP with Time Windows): Clients' locations have time windows within they must be served
- VRPPD (VRP with Pick Up and Delivery): Goods need to be moved from certain pickup clients to other delivery clients
- VRP LIFO (VRP with LIFO): Similar to the VRPPD, except an additional restriction is placed on the loading of the vehicles: at any delivery client, the item being delivered must be the item most recently picked up
- MDVRP (Multiple Depots VRP): Vehicles start from two or more depots
- PVRP (Period VRP): The VRP has to be solved for two or more days (in contrast to normal VRPs which work with one day routing)
- SDVRP (Split Delivery VRP): It is allowed that the same customer can be served by different vehicles
- VRPB (VRP with Backhauls): Customers can demand or return some commodities
- VRPSF (VRP with Satellite Facilities): Includes de use of satellite facilities to replenish vehicles during a route
- VRPSD (VRP with Stochastic Demands): Includes probabilistic descriptions of the demands to be served.

Notice that these are only a few pure VRPs but combinations of these could also appear in literature. Furthermore, the wide variety of cost functions adds more possibilities when a new VRP is modelled. Costs may include:

- Routes distances: These are clearly related both to the routing total time and the oil used to complete it
- Resources needed: The number and kinds of vehicles are a key issue, not only due to their direct costs but also for the human resources (drivers) needed
- Environmental issues: Transport is an important variable in environmental care not only for the atmospheric harm or the traffic jams but also due to noises, odours, etc.
- Intangible costs: Competence among enterprises, workers happiness or visual quality of solutions given, often are not considered but very important in decisions taking

3. RELATED WORK

The VRP has been studied for long and from several points of view. That is, different problem versions (see previous section) and different methodologies (e.g. artificial intelligence, operations research, simulation, etc) have been the aim of many researchers work. A recent survey of the VRP can be found at Sterling and Shapiro (1986).

Both the VRP and VRPTW problems have been formulated as an Integer Linear Programming (ILP) problem. A recent survey on the proposed formulations based on ILP is Letchford and Salazar-González (2006). Other complete formulations are those based on Column Generation (e.g. Desaulniers, Desrosiers and Solomon, (2005)) or Lagrangian Relaxation (e.g. Fisher and Jörnsten (1997)).

Since VRP is an NP-Hard problem, and hence, optimality prove becomes impossible. Therefore, several heuristic solution methods have been used. Construction methods like, for instance, the "Savings" method of Clarke and Wright (1964) have proved to work very well. There are other approaches to routes construction, like those including the "sweep" heuristic of Gillet and Miller (1974).

Local search methods have been deeply used too. These algorithms try to improve the current solution by moving to its neighbours depending on specifically defined "move operators". Two examples of these operators, commonly used together with constraint programming are Guided Local Search (Shaw, 1998) and Limited Discrepancy Search (Harvey and Ginsberg, 1995).

Metaheuristics go a step further, trying to avoid local minima. Many metaheuristics have been successfully applied to the VRP. Perhaps, the most important ones are Simulated Annealing (Bent and Van Hentenryck, 2004), Tabu Search (Cordeau, Laporte and Mecier, 2001), or Genetic Algorithms (Homberger and Gehring, 2005).

Given the complexity of real-world VRP problems, Constraint Programming complete search would be too inefficient to be considered by itself. Anyway, its combination with other local search methods can take advantage of a CP framework. The library presented in this paper allows experts on other techniques to combine their solutions with models written specifically for CP. Notice that all these methodologies often fall short to work with real-life VRP problems. Moreover, every technique has got its own drawbacks or weaknesses. Hybrid approaches seem to overcome single-methodology results (e.g. Juan, Faulin, Riera, Masip and Jorba (2009)).

4. THE VRP MODEL

4.1. Notation

For the model constraints and costs defined, the following notation has been used (notice that later, in the constraints definition, sometimes they may appear in lower case):

$C = C_1 \dots C_n$ are the customers to serve,
$M = M_1 \dots M_m$ are the available vehicles,
$Qm = Qm_1 \dots Qm_m$ are the vehicles capacities,
$V = V_1 \dots V_{n+2m}$ are the visits, with domain $1 \dots m$. Two sublists of V, F and L, are defined as the vehicles departure and arrival nodes.
For each visit V_i there exist the predecessor of that visit, P_i, and its successor, S_i.
R_i is the amount of goods to pick up in visit i.
After every visit, Q_i is the quantity of goods in the vehicle serving the visit.
Visit i is performed in time T_i.
In terms of costs, τ_{ij} and δ_{ij} represent the travel time and the distance from visit i to visit j, and τ_i is the time spent for pick up or delivery in i.

4.2. Constraints

The first approach to the library contains the following constraints:

4.2.1. Difference constraints

Equations (1) and (2) force predecessors and successors' lists to contain no repetitions. Thus, one client can have one and only one predecessor and successor.

$$\forall i, j \in V, i < j: p_i \neq p_j \quad (1)$$

$$\forall i, j \in V, i < j: s_i \neq s_j \quad (2)$$

4.2.2. Coherence constraints

Equations (3) and (4) connect the concepts successor and predecessor as follows: The former says that i is the successor of its predecessor. And the later says that i is the predecessor of its successor

$$\forall i \in V - F: s_{p_i} = i \quad (3)$$

$$\forall i \in V - F: p_{s_i} = i \quad (4)$$

4.2.3. Path constraints

Equations (5) and (6) are used to assure a route is visited by a single vehicle. Thus the vehicle assigned to i must be the same as those assigned to its predecessor and successor.

$$\forall\, i \in V - F: v_i = v_{p_i} \quad (5)$$

$$\forall\, i \in V - L: v_i = v_{s_i} \quad (6)$$

4.2.4. Capacity constraints

Equations (7) and (8) count the goods picked up in a route, and the goods delivered in that route. Thus, the first constraint says the goods accumulated after visiting client i is the addition of those accumulated in its predecessor plus those taken in i. The second is similar but using the successor.

$$q_i \geq 0$$

$$r_i \neq 0$$

$$\forall i \in V - F: q_i = q_{p_i} + r_i \quad (7)$$

$$q_i = q_{s_i} - r_{s_i} \forall i \in V - L \quad (8)$$

Furthermore, the maximum capacity defined for a vehicle Qm_j must limit the accumulated capacities for every client visited by that vehicle. This can be done either with an extra constraint or with domains bounding.

4.2.5. Time constraints

Equations (9) and (10) bound the accumulated time spent by a vehicle visiting client i. This time is at least, the accumulated time in the predecessor of i, plus the time spent in i. Equally, this time must be, at most, the accumulated time in the successor of i, minus the time spent in i.

$$t_i \geq 0$$

$$t_i \geq t_{p_i} + \tau_{p_i,i} \forall i \in V - F \quad (9)$$

$$t_i \leq t_{s_i} - \tau_{i,s_i} \forall i \in V - L \quad (10)$$

4.2.6. Time constraints 2

Equations (11) and (12) are similar to equations (9) and (10) including travel times from and to the predecessor and the successor of *i* respectively.

$$t_i \geq 0$$

$$t_i \geq t_{p_i} + \tau_i + \tau_{p_i,i} \forall i \in V - F \quad (11)$$

$$t_i \leq t_{s_i} - \tau_i - \tau_{i,s_i} \forall i \in V - L \quad (12)$$

4.3. Cost functions

4.3.1. Distance travelled

Equations (13) and (14) model the cost as the additions of the travelled distances.

$$d = \sum_{i \in V-F} \delta_{p_i,i} \quad (13)$$

$$d = \sum_{i \in V-L} \delta_{i,s_i} \quad (14)$$

5. THE VRP LIBRARY

In order to build a model using this library, variables for P, S, V, C, Q and T must be defined. Furthermore, input lists or matrices defining costs are needed.

Once the variables and domains are declared, the following rules have been programmed in ECLiPSe and can be called to model the constraints of the problem:

5.1. Programmed constraints

5.1.1. Difference constraints

constraintsDifferent(+P,+S,+CN)

5.1.2. Coherence constraints

constraintsCoherence(+P,+S)

5.1.3. Path constraints

constraintsPath(+V,+P,+S,+MN)

5.1.4. Capacity constraints

constraintsCapacity(+Q,+R,+P,+S,+MN)

5.1.5. Time constraints

constraintsTime(+T,+Times,+P,+S,+MN)

5.1.6. Time constraints 2

constraintsTime2(+T,+Times,+P,+S,+MN)

5.2. Programmed cost functions

5.2.1. Distance travelled

costFunctionTotalDistanceMin(+T,+MN,-Cost)

Notice that other constraints to remove symmetries or specifications of the above constraints are already programmed and documented in the library. Moreover, new constraints and costs are being continuously added.

6. EXAMPLE OF CVRP USING THE LIBRARY

As said before, the library is written in ECLiPSe (2009). In order to write a model, the library (stored in the same directory as the model) is loaded as shown in Fig. 1.

```
:-['vrp_lib.ecl'].
```

Figure 1: VRP Library Call

Furthermore, the Constraint Satisfaction Problem (i.e. variables, domains and constraints) must be declared. Later the search is performed and the solution is shown. The set of necessary calls can be seen in Fig. 2.

```
solve(B, H1,H2):-
variables(VARS,VARSN),
domains(VARS,VARSN),
constraints(VARS,VARSN),
optimise(VARS,VARSN,B,H1,H2),
showSolution(VARS,B).
```

Figure 2: General Rule to Solve a CSP with the VRP Library

Notice that H1 and H2 are the heuristics to perform the search. These are: the order the variables are assigned, and the order in which the possible values of a variable are chosen, respectively.

The rule *variables* (see Fig. 3) contains the definition of the necessary variables. These are usually lists, since they store one variable for each vehicle, customer, etc.

```
variables([C,M,V,Q,T,Times,R,Qv,P,
S],[CN,MN]):-
readCustomers(C,CN,MN,R,Times),
readVehicles(M,MN,Qv),
VN is CN + 2 * MN,
P_SN is CN + MN,
length(V,VN),
length(Q,VN),
length(T,VN),
length(P,P_SN),
length(S,P_SN).
```

Figure 3: CSP's Variables Definition

Notice that rules *readCustomers* and *readVehicles* are used to read the problem instance to be solved.

The rule *domains* (see Fig. 4) defines the feasible values the variables of the CSP might take. Normally, the smaller the domains are, the faster the search will be.

```
domains([C,M,V,Q,T,_,_,Qv,P,S],[CN
,MN]):-
(foreach(Ci,C),for(I,1,CN) do
 Ci #= I),
(foreach(Mi,M),for(J,1,MN) do
 Mi #= J),
decompose(V,MN,_,_,V_FL,_,_),
V_FL::M,
max(Qv,Max),
Q::0..Max,
T::0..500,
length(P,P_SN),
P :: 1..P_SN,
S :: 1..P_SN.
```

Figure 4: CSP's Domains Definition

The rule *constraints* (see Fig. 5) is a set of calls to the library constraints which must be included in the model.

```
constraints([_,_M,V,Q,T,Times,R,Qv
,P,S],[CN,MN]):-
constraintsCiclicRoutes(V,MN),
constraintsDifferent(P,S,CN),
constraintsCoherence(P,S),
constraintsPath(V,P,S,MN),
constraintsCapacity(Q,R,P,S,MN),
constraintsMaxCapacity(Q,V,Qv,MN),
constraintsTime2(T,Times,P,S,MN),
constraintsSymmetriesVisitsSorted(
V,MN),
constraintsSymmetriesAllVehiclesUs
ed(P,S,CN,MN).
```

Figure 5: CSP's Constaints Definition

Finally, the rule *optimise* (see Fig. 6) contains the necessary calls to optimise the model.

```
optimise([C,_,V,Q,T,_,_,_,P,S],[_,
MN],B,H1,H2):-
flatten([P,S],VARSflat),
flatten([V,C,Q,T],VARS2flat),
costFunctionTotalDistanceMin(T,MN,
Cost),
bb_min((search(VARSflat,0,H1,H2,co
mplete,[backtrack(B)]),indomain(Co
st)),Cost,bb_options{strategy:cont
inue}), once(labeling(VARS2flat)).
```

Figure 6: CSP's Search Definition

In order to perform the search, only predecessors and successors are instantiated. The rest of values of the model become either instantiated or bounded depending on propagation.

Notice that new improvements in models specification and search might be included after publishing this paper. The search presented is the basic one. Download the example files from the URL introduced in section 7 in order to take advantage of new advances and to avoid problems with new versions.

7. CASE STUDY

In order to test the library, the following CVRP has been programmed:

$C = [C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8]$
$M = [M_1, M_2, M_3]$
$Qm = [220, 220, 220]$
$R = [69, 80, 87, 38, 54, 122, 74, 91]$

While the depot is placed in the origin *(0,0)*, the 8 clients are respectively *in (-49.95, -95.30), (-70.61, -13.54), (-1.01, 76.81), (-5.94, 20.97), (9.74, 18.50), (-70.42, 10.00), (95.50, -90.35), (35.32, 59.99).*

For this example, in terms of cost, only travel times will be considered. Let us suppose, for simplification, that the travel time is defined as the integer part of the Euclidean distance between two points:

$$\tau = \begin{pmatrix} 0 & 84 & 178 & 124 & 128 & 107 & 145 & 155 \\ 84 & 0 & 114 & 73 & 86 & 23 & 183 & 81 \\ 178 & 114 & 0 & 56 & 59 & 96 & 193 & 38 \\ 124 & 73 & 56 & 0 & 15 & 65 & 150 & 48 \\ 128 & 86 & 59 & 15 & 0 & 80 & 138 & 61 \\ 107 & 23 & 96 & 65 & 80 & 0 & 193 & 61 \\ 145 & 183 & 193 & 150 & 138 & 193 & 0 & 199 \\ 155 & 81 & 38 & 48 & 61 & 61 & 199 & 0 \end{pmatrix}$$

Given this CVRP, after generating the CSP (using the library presented) the output of the ECLiPSe application is shown in Fig. 7.

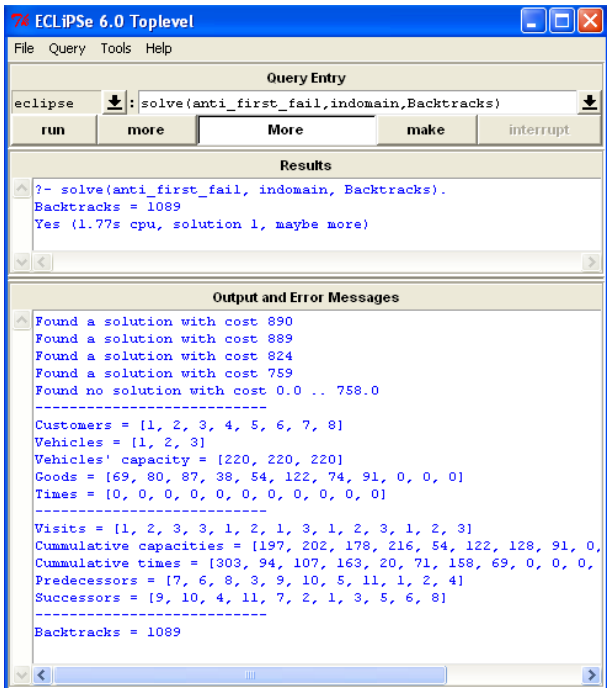

Figure 7: Result of CSP Model Optimisation

For better comprehension, the solution found is painted in Fig. 8.

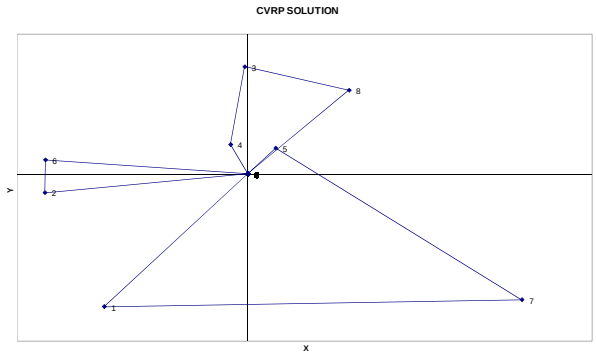

Figure 8: Optimal Routes Found

8. CONCLUSIONS AND FUTURE WORK

The aim of the presented work is to ease researchers and students working around the VRP their construction of models using constraint programming. The library is currently in an initial stage and only a few constraints

and cost functions are available. On the one hand, we are working on adding first the most common constraints. On the other hand we are proposing new cost functions related to environmental and non tangible issues.

The idea is to include new constraints and costs to the library, completing it to allow the modelling of as much kinds of VRPs as possible. The library is available at the URL http://gres.uoc.edu/VRP. We expect to complete it thanks to comments and questions sent.

ACKNOWLEDGMENTS

The research of this paper has been partially sponsored by the Internet Interdisciplinary Institute (IN3) and the Knowledge Community on *Hybrid Algorithms for solving Real-life rOuting, Scheduling and Availability problems* (*HAROSA*).

REFERENCES

Tsang, E., 1993. Foundations of Constraint Satisfaction. Academic Press.

Rossi, F., Van Beek, P., Walsh, T. 2006. Handbook of Constraint Programming. *Foundations of Artificial Intelligence Series*, Elsevier, Amsterdam (The Nederlands).

Golden, B., Raghavan, S., Wasil, E. 2008. The vehicle routing problem. *Latest advances and new challenges*, Springer, New York (USA).

Sterling, L., Shapiro, E. 1986. The art of Prolog. *The MIT Press*. Cambridge (USA).

Letchford, A.N., Salazar-González, J.J. Projection results for vehicle routing. Mathematical *Programming,* 105(2):251-274, 2006.

Desaulniers, G., Desrosiers, J., Solomon, M.M. (editors) Column Generation. Springer, 2005.

Fisher, M.L., Jörnsten, K.O. Vehicle Routing with Time Windows: Two Optimization Algorithms. *Operations Research*, 45(3):488-492, 1997.

Clarke, G. and J. Wright. Scheduling of Vehicles from a central Depot to a Number of DeliveringPoints. *Operations Research,* 12, 568-581, 1964.

Gillet, B., Miller, L.R. A Heuristic Algorithm for the Vehicle Dispatch Problem. *Operations Research*, 22:340-349, 1974.

Shaw, P. Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. *Fourth International conference of Principles and Practice of Constraint Programming (CP'98)*, 1998.

Harvey, W.D., Ginsberg, M. L. Limited Discrepancy Search. *Proceedings of the Fourteenth International Joint conference on Artificial Intelligence (IJCAI-95),* volume 1, pg. 607-615, Montreal (Canada), 1995.

Bent, R., Van Hentenryck, P. A two-stage Hybrid Local Search for the Vehicle Routing Problem with Time Windows. *Transportation Science*, 38(4):515, 2004.

Cordeau, J.F., Laporte, G., Mecier, A. A Unified Tabu search Heuristic for Vehicle Routing Problems with Time Windows. *Journal of the Operations Research Society*, 52(8):928-936, 2001.

Homberger, J., Gehring, H. A two-phase Hybrid Metaheuristic for the Vehicle Routing Problem with Time Windows. *European Journal of Operation Research*, 162(1):220-238, 2005.

Juan, A.A., Faulin, J., Riera, D., Masip, D., Jorba, J. A Simulation-based Methodology to assist Decision-makers in real Vehicle Routing Problems. *Proceedings of the 11th Int. Conf. on Enterprise Information Systems*. Milan, 2009.

Toth, P. and D. Vigo. 2002. The Vehicle Routing Problem. *SIAM Monographs on Discrete Mathematics and Applications.* SIAM

ECLiPSe Home. *The ECLiPSe Constraint Programming System.* Available from: http://87.230.22.228/index.html [accessed 2 June 2009]