Improving Quality and Efficiency in Home Health Care: an application of Constraint Logic Programming for the Ferrara NHS unit*

Massimiliano Cattafi[1], Rosa Herrero[2], Marco Gavanelli[1], Maddalena Nonato[1], Federico Malucelli[3], and Juan José Ramos[2]

1 **ENDIF, Università di Ferrara, Italy**
{massimiliano.cattafi, marco.gavanelli, maddalena.nonato}@unife.it
2 **Dept. de Telecomunicació i Enginyeria de Sistemes, UAB, Spain**
rherrero.math@gmail.com, juanjose.ramos@uab.cat
3 **Dipartimento di Elettronica e Informazione, Politecnico di Milano, Italy**
malucell@elet.polimi.it

Abstract

Although sometimes it is necessary, no one likes to stay in a hospital, and patients who need to stay in bed but do not require constant medical surveillance prefer their own bed at home. At the same time, a patient in a hospital has a high cost for the community, that is not acceptable if the patient needs service only a few minutes a day.

For these reasons, the current trend in Europe and North-America is to send nurses to visit patients in their home: this choice reduces costs for the community and gives better quality of life to patients. On the other hand, it introduces the combinatorial problem of assigning patients to the available nurses in order to maximize the quality of service, without having nurses travel for overly long distances.

In this paper, we describe the problem as a practical application of Constraint Logic Programming. We first introduce the problem, as it is currently addressed by the nurses in the National Health Service (NHS) in Ferrara, a mid-sized city in the North of Italy. Currently, the nurses solve the problem by hand, and this introduces several inefficiencies in the schedules.

We formalize the problem, obtained by interacting with the nurses in the NHS, into a Constraint Logic Programming model. In order to solve the problem efficiently, we implemented a new constraint that tackles with the routing part of the problem. We propose a declarative semantics for the new constraint, and an implementation based on an external solver.

1998 ACM Subject Classification D.3.2 Constraint and logic languages

Keywords and phrases CLP(FD), Nurse Scheduling Applications, Home Health Care

Digital Object Identifier 10.4230/LIPIcs.ICLP.2012.415

1 Introduction

One of current trends to reduce costs and maintain service quality of health services is to close peripheral hospitals, reduce patients hospitalization, and concentrate the service at

* We thank Andrea Peano for his help with the significance tests. This work was partially supported by EU project *ePolicy*, FP7-ICT-2011-7, grant agreement 288147. Possible inaccuracies of information are under the responsibility of the project team. The text reflects solely the views of its authors. The European Commission is not liable for any use that may be made of the information contained in this paper.

Technical Communications of the 28th International Conference on Logic Programming (ICLP'12).
Editors: A. Dovier and V. Santos Costa; pp. 415–424

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

few, big structures, able to provide specialized treatments and high quality consultancy. At the same time, though, those patients who do not need to be treated in a hospital, must be provided health care directly at their homes. The challenge is to keep costs at a low level, while achieving high service quality standards, comparable to those achievable at a hospital.

In this paper we describe an application concerning home health care in the city of Ferrara, Italy. We describe the problem as stated by the workers in the local agency of the National Health Service (NHS), together with the data they provided us. Then, we model the problem in Constraint Logic Programming on finite domains (CLP(FD)).

1.1 The home health care service in Ferrara

At present, the home health care (HHC) service in the city of Ferrara, Italy, is managed by the local agency of the National Health Service (NHS), namely AUSL 109. All patients who are not self sufficient and in need for medical treatment are eligible for HHC. Each request is thus characterized by a patient identifier (name and address), a medical treatment, and the specific day of the week when the treatment must be delivered (each patient can have more than one request per week).

Service is provided by a set of qualified nurses. Every day, each nurse starts his/her duty at the city hospital, visits the patients in his/her list, delivers the required treatments and travels by car from one patient's home to the next, until (s)he finally returns to the hospital.

A treatment lasts from 5 to 60 minutes, depending on its specific characteristics.

While Ferrara is a medium-size town (about 150,000), the area administered by AUSL 109 is rather large and its population ageing. Although most of the population is concentrated in town, a number of elderly people live in the countryside and they are those more likely to be enrolled in the service. Therefore, the service is characterized at the same time by a high variance of duration and a significant geographical dispersion of the requests.

Scheduling such a service poses several challenges; good solution should achieve:

- from the NHS point of view, the minimization of the travel time over the service time; in fact during travel a nurse is on duty but is not delivering any service.
- from the nurses point of view, the equidistribution of the workload, which can not be guaranteed by simply equally subdividing patients, due to heterogeneous requests.
- from the patient point of view, a good degree of *loyalty*, i.e., the number of different nurses who are in charge of a single patient should be kept as low as possible.

A total of 15 nurses is involved. A duty should last up to 7 hours and 12 minutes. As a representative sample, in February 2010 there were 3323 requests, subdivided among 458 patients.

At present, nurses organize their duties themselves. In order to simplify the subdivision of the patients to the nurses, the territory pertaining AUSL 109 has been partitioned statically into 9 zones Each nurse receives in charge most of the patients belonging to one such area. Then the nurse tries to fit the patients requests into the working shifts while complying with the maximum workload allowed. Such decisions are not driven by any optimization criteria, and the routing is not necessarily optimal within the day, leaving apart what could be gained in terms of travelled distance if requests would be exchanged between nurses. Due to the greedy procedure followed, the nurse weekly schedules have very different workloads and balancing this load over the months leads to a detriment in loyalty. Nurses complain about such disparities, and have difficulties adapting their schedule to new incoming patients, new treatments, or to any other change. Moreover, if the workload balance could be improved by optimizing the routing component, nurses could be available at the hospital for others tasks,

thus reducing the overall costs. In addition, an improved routing plan would impact on the direct expenses related to gas and car usage, which contribute to the overall cost.

2 A Constraint for the Traveling Salesman Problem

Before delving into the actual CLP model of the problem, we present a new constraint useful to address efficiently the routing component of the problem. Intuitively, the new constraint provides the length of the shortest Hamiltonian cycle connecting a given subset of the nodes in a graph.

Given a fully-connected graph $G \equiv (N, E)$ with a special node $0 \in N$, a weighting function $d : N \times N \mapsto \mathbb{R}$ (also represented in matrix notation $D = (d_{i,j})$), and a selection function $s : N \mapsto \{0, 1\}$ (also represented as a list, or a 1-dimensional matrix S) constraint

$$traveltime(N, D, S, T^{trv}) \tag{1}$$

solves a TSP and computes the length of the shortest Hamiltonian cycle associated to the set of nodes $N' = \{n \in N | S(n) = 1\}$. More precisely, given a list of nodes $N = (p_1, p_2, \ldots, p_m)$, a matrix of distances D, and a list of values $S = (s_1, \ldots, s_m)$, constraint (1) is true iff

$$T^{trv} = \min_{Path} \sum_{(i,j) \in Path} d_{i,j}$$

such that

- $Path$ is a sequence of the form

$$0, (0, p_{k_1}), p_{k_1}, (p_{k_1}, p_{k_2}), p_{k_2}, \ldots (p_{k_{n-1}}, p_{k_n}), p_{k_n}, (p_{k_n}, 0), 0$$

  that alternates nodes with edges, starting and ending at node 0;
- the visited nodes are exactly those corresponding to elements in the list S:

$$p_i \in Path \iff S_i = 1.$$

Pseudocode in Figure 1 outlines the implementation of the *traveltime* constraint. Operationally, it awakes every time one of the s_i variables is instantiated. If $s_i = 1$, it means node p_i must be visited. Note that, in a generic node of the search tree, some of the s_i variables will have value 1, some will have been set to 0, and some will still be unassigned. The predicate in line 2 selects in $DefinitelyVisited$ the nodes that have definitely to be visited. This variable is, in turn, passed as an argument to the predicate which computes the corresponding TSP (line 3). The TSP thus takes into account at this stage the nodes which are currently known to be visited, i.e., those for which the S variable has been set to 1.

If not all of the S variables are ground (test on line 4), the TSP cost provides a valid lower bound to the actual travel time (line 6) and can be used in a branch-and-bound search. In fact, if the lower bound is higher than the elements in the domain of the variable T^{trv} (for example, because the working hours of the nurse are almost all used for servicing the patients, so there is not enough time for traveling), we can immediately fail and backtrack, avoiding to continue the search in a wrong branch of the search tree.

When all of the S variables are ground, the cost of the TSP becomes the real travel time, and we are able to finally fix the value of T^{trv} to the TSP cost (line 5).

```
1  traveltime(P,D,S,T^trv):-
2     select_definitely_visited_nodes(P,S,DefinitelyVisited),
3     compute_tsp(DefinitelyVisited,D,LowerBound),
4     (ground(S)
5        -> T^trv = LowerBound
6        ;  T^trv ≥ LowerBound,
7           suspend(traveltime(P,D,S,T^trv))
8     ).
9  select_definitely_visited_nodes([],[],[]).
10 select_definitely_visited_nodes([P_i|P],[S_i|S],Definitely):-
11    (ground(S_i), S_i=1 -> Definitely = [P_i|LV] ; Definitely = LV),
12    select_definitely_visited_nodes(LP,LS,LV).
```

Figure 1 Pseudocode for the *traveltime* constraint.

3 A Constraint Logic Programming Model

Formally, the input data consists of:

- a set $\mathcal{S}_{serv}$ of services, of size N_s; for each service s we know the patient pat_s, the day day_s and the duration dur_s
- a matrix of distances D; the element $d_{i,j}$ is the travel time from patient i to patient j (if i and j are both greater than 0), or from/to the hospital (if $i = 0$ or $j = 0$)
- $\mathcal{S}_{nurse} = \{1, \ldots, N_n\}$ is the set of available nurses
- N_d is the number of days considered in the scheduling
- MpD is the amount of minutes available per day for each nurse; this includes both service time and travel time

A solution is an assignment of a nurse to each service, respecting all the constraints. The quality of the solution depends on how balanced the week workloads of the nurses are and on how many different nurses take care of the same patient during the week.

3.1 The CLP Model

To each service s we associate a decision variable $Nurse_s$ that can take a value between 1 and the number of available nurses N_n.

It can be useful to represent the nurses variables also in their Boolean channeling version, using constraint reification; this simplifies the definition of some requirements, as will be clear in the following. We have a matrix SN of size $N_s \times N_n$ such that

$$SN_{s,n} = 1 \iff Nurse_s = n. \qquad (\forall s \in \mathcal{S}_{serv}, \forall n \in \mathcal{S}_{nurse}) \tag{2}$$

We are interested in computing the workload of each nurse n in each day d: $DayWL_{n,d}$. Each day workload is the sum of the total service time and the travel time of that nurse:

$$DayWL_{n,d} = T^{svc}_{n,d} + T^{trv}_{n,d} \qquad (\forall n \in \mathcal{S}_{nurse},\ \forall d \in 1 \ldots N_d). \tag{3}$$

The total day workload for each nurse cannot exceed MpD, so for each day d and each nurse n, the domains of variables $DayWL_{n,d}$, $T^{svc}_{n,d}$ and $T^{trv}_{n,d}$ are from 0 to MpD.

The week workload $WeekWL_n$ of nurse n is the sum of the respective day workloads

$$WeekWL_n = \textstyle\sum_{d=1}^{N_d} DayWL_{n,d} \qquad (\forall n \in \mathcal{S}_{nurse}). \tag{4}$$

The service time is the total time of the durations of the services given by nurse n in day d:

$$T^{svc}_{n,d} = \sum_{s \in \mathcal{S}_{serv}, day_s = d} dur_s \cdot SN_{s,n}. \tag{5}$$

As mentioned in Section 2, the routing part is addressed by constraint *traveltime* (Eq. 1) that solves the TSP of a nurse that visits a subset of the patients. In order to compute the travel time $T^{trv}_{n,d}$ of nurse n in day d, we need to provide to such constraint

1. the nodes of the graph, that are the patients' locations
2. the matrix of distances D,
3. the selection function S (in its list representation), that is a sub-matrix of the SN matrix,
4. and the (finite domain) variable that represents the travel time: $T^{trv}_{n,d}$.

More precisely, since we want to compute the travel time for day d, item 1 will be the set $Patients^d \triangleq \{pat_s | s \in \mathcal{S}_{serv}, day_s = d\}$ of those patients that will be visited in day d, while item 3 will be the sub-matrix $SN^d_n \triangleq \{SN_{s,n} | s \in \mathcal{S}_{serv}, day_s = d\}$ containing only those services to be given in day d. In other words, the actual parameters passed to constraint *traveltime* to compute the traveltime of nurse n in day d will be:

$$traveltime(Patients^d, D, SN^d_n, T^{trv}_{n,d}).$$

3.2 The Objective Function

The requirements given by the chief nurse are to optimize two main objectives, namely the equal repartition of the workload to the various nurses and the loyalty, although psychological factors or tiredness can also affect the quality of the service.

Concerning the first objective, there are various ways to achieve balanced week workloads for the nurses [14]. We chose to minimize the maximum week workload, obtained as

$$MaxWeekWL = \max_{n \in \mathcal{S}_{nurse}} WeekWL_n$$

One way to obtain maximum loyalty is to minimize the number of nurses that visit a same patient. Let $ServicePat_p$ be the set of services of patient p. The information if a patient p is visited during the week by nurse n is given by:

$$PN_{p,n} = \bigvee_{s \in ServicePat_p} SN_{s,n} \qquad \forall p \in \mathcal{S}_{patient}, \forall n \in \mathcal{S}_{nurse}$$

(where we identify truth values *true* and *false* with 1 and 0, respectively); then

$$LoyaltyPenalty = \sum_{p \in \mathcal{S}_{patient}, n \in \mathcal{S}_{nurse}} PN_{p,n}$$

The global objective can be given as a weighted sum of the two components

$$\min(\alpha_1 \cdot MaxWeekWL + \alpha_2 \cdot LoyaltyPenalty), \tag{6}$$

where α_1 and α_2 are positive real numbers that can be chosen by the user in order to reflect the current priorities adopted in the AUSL. Of course, such values can be tuned later on.

4 Example

As an example, we have three patients, requesting a total of 5 services, whose durations are in Fig 2 and the distance matrix (in upper triangular form) is in Figure 3. Assume we have two nurses, n_1 and n_2, and that the limit on the day workload is $MpD = 30$.

One solution could be to assign

patient	mon	thu
p_1	10	5
p_2		20
p_3	20	5

Figure 2 Patients' requests with durations.

	h	p_1	p_2	p_3
h		3	3	5
p_1			2	7
p_2				8

Figure 3 Distance Matrix.

- on *mon*, nurse n_1 to patient p_1 (formally, $Nurse_{(p_1,mon)} = n_1$) and nurse n_2 to p_3
- on *tue*, nurse n_1 to patients p_1 and p_3, and nurse n_2 to p_2.

In this assignment, we have $DayWL_{n_1,mon} = T^{svc}_{n_1,mon} + T^{trv}_{n_1,mon} = 10 + (3 + 3) = 16$ (going from the hospital h to p_1 and coming back); $DayWL_{n_2,mon} = 20 + (5 + 5) = 30$; $DayWL_{n_1,tue} = (5 + 5) + (3 + 8 + 5) = 26$; $DayWL_{n_2,tue} = 20 + (3 + 3) = 26$. The total week workload is $WeekWL_{n_1} = 16 + 26 = 42$ for nurse n_1 and $WeekWL_{n_2} = 30 + 26 = 52$ for n_2. The loyalty penalty will be 1 for patients p_1 and p_2 (that are visited by one nurse) and 2 for p_3, that is visited by both nurses. So, the value of the objective function will be $\alpha_1 \cdot \max\{42, 52\} + \alpha_2 \cdot (1 + 1 + 2) = 52\alpha_1 + 4\alpha_2$.

5 Implementation

The TSP solving algorithm (predicate `compute_tsp` in Figure 1) can be implemented in CLP(FD), with different constraint models.

One model assigns a decision variable for each city to be visited. We have a sequence of decision variables $X_1, \ldots, X_n$, each of them ranging on the set of cities to be visited, and where X_1 is the first city to be visited, X_2 is the second, $\ldots$, X_n is the last city to be visited. The fact that all cities must be visited is imposed through an `alldifferent` constraint [13]. In this model, the cost is the sum of the distances $d(X_1, X_2) + d(X_2, X_3) + \cdots + d(X_{n-1}, X_n) + d(X_n, X_1)$.

A second model uses the `circuit` constraint [4] for which various propagation algorithms have been proposed [6, 10]. Again, we have a sequence of decision variables $X_1, \ldots, X_n$, each ranging on the set of possible cities, but in this case the meaning is different: X_1 is the city to be visited immediately after city number 1, X_2 is the city to be visited after city whose name is "2", $\ldots$, X_n is the city that is visited after the city named n. The `circuit` constraint ensures that allowed assignments form a Hamiltonian circuit, and the cost is the sum $d(1, X_1) + d(2, X_2) + \cdots + d(n, X_n)$.

However, it is well known in the literature [6] that solving large TSPs in CLP(FD) is very demanding in terms of computing time, so we decided to implement predicate `compute_tsp` as an invocation of an efficient TSP solver [9], based on the Lagrangian Relaxation technique used in Operations Research. We could have used other solvers, but we found that our choice performed well on the typical size of the considered TSP instances. Although the TSP instances were very difficult for a CLP(FD) implementation, they were rather easy for Lagrangian Relaxation, and solving them through LR did not show a deterioration in performance with respect to state-of-the-art TSP solvers [1], so we preferred to use a solver for which we had access to the source code. A detailed description of the Lagrangian Relaxation technique is out of the scope of this work; the interested reader can refer to [9].

6 Search Strategies

We tried our model with five search strategies. The first four were developed with the goal of obtaining a good general-purpose search strategy; then we tried to improve by exploiting better the structure of the problem.

The **Generic Search (GS)** strategy is a depth-first search in which the next variable to be assigned is selected with the smallest domain heuristic. Since the decision variables are the $Nurse$ variables (Section 3.1), we select first the service that can be assigned to the smallest number of nurses. The assigned nurse is selected at random. The **Generic Search with Restarts (GSR)** also applies restarts with the optimal timeout sequence [11].

We also modified the variable selection heuristics; instead of selecting the variable with the smallest domain within the services of the whole week, we try to complete the assignments of a single day before starting with another day (first assign all the services of the first day, then the second day, etc). The idea is that if we made some wrong decisions in one day, so that it is impossible to assign all the patients of that day, we want to fail as soon as possible before initiating the assignments in other days. Within each day, we select first the variable with the smallest domain. This strategy was applied without restarts, **Generic Search by Day (GSD)** and with restarts **Generic Search by Day with Restart (GSDR)**.

Finally, we defined a search strategy more tailored to the problem at hand, called **Loyalty Guided Search (LGS)**. We first sort the services in decreasing order of duration, so that those services with higher duration will be assigned first. The idea is to try to fit first the most difficult services into the available time for the nurses. Then, given a service s of patient pat_s, we try to assign him/her a nurse who is already visiting this patient, in order to minimize the $LoyaltyPenalty$. The domain of $Nurse_s$ is divided into two parts: the nurses who are already visiting this patient and those who are not; we try first the first part, then the second. Moreover, both parts are sorted by the current $WeekWL$ of the nurse; in this way, we try first the nurses that are less occupied, in order to balance the workload.

7 Experiments and Results

The program was implemented in the open-source CLP system ECL^{i}PS^e 6.0 [2], and linked to a Java part implementing Lagrangian Relaxation (LR) for the TSP. All tests were performed on an Intel i5 processor 2.40GHz computer with 4GB of RAM on four weekly instances.

Figure 4 shows the computation time required by the routing aspect of the problem with the various methods described in Section 5. The values are obtained by imposing a full weekly assignment of services to nurses and then computing the $N_n \times N_d$ resulting TSPs with each of the different methods. It can be noticed that the circuit-based one is more efficient than the alldifferent-based one, however solving the TSPs with Lagrangian Relaxation is orders of magnitude faster than both of them (times are reported on a logarithmic scale). Using CLP(FD) as a unifying framework, it is practical and convenient to take advantage of the efficiency of LR on this specific subproblem by enclosing it in our *traveltime* constraint.

Table 1 shows the best results obtained for the five search strategies described in Section 6 running them for a maximum of 10 minutes. The randomized algorithms (the GS* strategies) were run 20 times each. For each week, we show the Objective and the corresponding $MaxWeekWL$ and $LoyaltyPenalty$. The Objective is given by the weighted sum in Eq. 6; in these particular experiments, we used $\alpha_1 = \alpha_2 = 1$, so Objective is simply the sum of the two subsequent columns. The results are compared to the solution hand-made (HMS) by the nurses considering the division of the city into 9 areas. We can see that the model was very effective, as all the search strategies were able to improve on the hand-made

Table 1 Best results for each search strategy.

	First Week	Second Week	Third Week	Fourth Week
	Objective=WL+LP	Objective=WL+LP	Objective=WL+LP	Objective=WL+LP
GS	2203 1918 + 285	2371 2064 + 307	2331 2033 + 298	2387 2063 + 324
GSR	2125 1841 + 284	2347 2040 + 307	2270 1963 + 307	2345 2022 + 323
GSD	2185 1905 + 280	2351 2052 + 299	2255 1963 + 292	2389 2071 + 318
GSDR	2097 1811 + 286	2345 2033 + 312	2263 1958 + 305	2342 2011 + 331
LGS	1982 1782 + 200	2277 2042 + 235	2181 1954 + 227	2290 2034 + 256
HMS	2356 2124 + 232	2405 2153 + 252	2395 2141 + 254	2433 2146 + 287

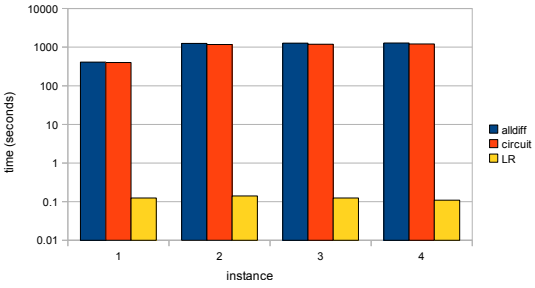

Figure 4 Comparison of computing time required to solve the TSPs with the different methods.

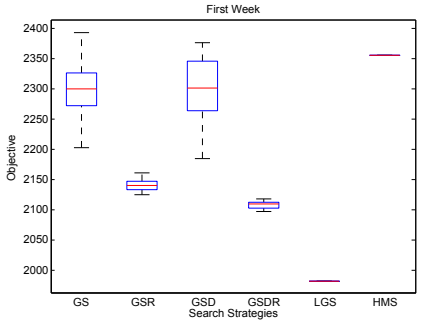

Figure 5 Box plot of the 5 strategies and the hand-made solution (HMS).

solution. Moreover, LGS was able to improve on the hand-made solution both in terms of equidistribution of the workload and in terms of loyalty, thus improving both the working conditions of the nurses and the service quality for the patients. Unluckily, we were not able to compute the optimal solution, so we cannot compare with it.

Since some of the search strategies use randomization, we also show the box plots of the 20 repetitions (Figure 5). The plots reveal that restarts are the most important factor in the general purpose strategies. The strategies not using restarts often were unable to improve the hand-made solution. Labeling on single days can sometimes give a further improvement. However, a search strategy tailored for the problem is able to provide a large improvement with respect to the general purpose ones. Notice that LGS and HMS are strategies that do not include randomization, so they always provide the same solution each time they are executed; this explains why the box plot reduces to a single line. A significance test supports the conjecture that LGS improves upon the hand made solution. The Wilcoxon-Mann-Whitney test [3] rejects the hypothesis that LGS is worse than HMS with a p-value of 0.01429, well below the usual significance threshold of 0.05.

8 Related Work

The efficient delivery of HHC service attracted the attention of the CLP and the Operations Research communities. Application papers are generally focused on the particular type of service that has to be optimized. In many countries for example the Home Health Care service is managed together with the Home Care that involves other types of services and

most of the times requires the specification of time windows in which the services have to be delivered and this is one of the main differences that we found with our case.

For example, Bertels and Fahle [5] adopt a hybrid approach, combining Constraint Programming, local search and Linear Programming. The approach takes advantage of the presence of tight time windows: *"Due to time window constraints, in the HHC only few permutations correspond to feasible orderings. In our approach we enumerate those orderings by a CP approach, and we use an LP to find optimal start times ..."*. In our case time windows are not present thus enumerating the feasible orderings is not viable.

Laps Care [7] is a system adopted in Sweden for Home Care, although it is also able to consider some of the issues in HHC. Laps Care uses an iterative method, in which an initial solution uses a single route for each service; then routes are joined until no further improvement is possible. To escape from the local optimum, one of the routes is split into one route for each patient, and the joining phase restarts.

Looking at the problem from a more abstract viewpoint, one may see some similarities with the classical Capacitated Vehicle Routing Problem (CVRP). In the CVRP a set of disjoint routes for a fleet of vehicles has to be found so that all customers (nodes) are visited, the required quantity of goods is delivered to each customer, the capacity of the vehicles is not exceeded and the objective function is minimized. The usual objective function is the overall traveled distance or the number of vehicles. In our case we can see nurses as vehicles and patients as customers. There are some important differences with CVRP that make all the efficient method developed for the classical problem not applicable in our case. One difference concerns the capacity. As in CVRP we may consider nurse daily duty time as a capacity constraint, however, unlike the CVRP, in our case the sequence in which patients are visited matters on how the capacity is consumed. This actually turns our problem into a time constrained VRP which is not as easy as the CVRP and for which the classical CVRP methods are not so efficiently adapted.

The other difference concerns the objective function. On the one hand, as in VRP, we would have to minimize the total traveled distance, in order to make the service as efficient as possible, on the other hand we have to balance the workload among nurses. Thus this component of the objective function is a kind of bottleneck (min-max), that is difficult to address with OR methods. Finally the loyalty factor is component of the objective function that makes our problem very peculiar, not to say unique.

Various works consider how to solve the TSP, or its variant with Time Windows, in CP or with hybrid algorithms [6, 12, 8]; the TSP is only a component of the HHC problem.

9 Conclusions

In this paper, we presented a Constraint Logic Programming application for a Home Health Care problem. We modelled the problem that is currently solved by hand by the nurses of the National Health Service unit of the city of Ferrara, in Italy. The novelty of the model stands in two issues that are peculiar of the problem in Ferrara. The first issue is the objective: reducing the disparities in workload of the nurses, while at the same time improving the quality of service from the patients' viewpoint, by keeping minimal the number of different nurses that take care of a same patient. The second issue is the implementation of a new constraint that addresses the routing component of the problem. The constraint was implemented by embedding into a constraint an efficient solver for the Travelling Salesperson Problem. Although the new constraint is implemented through a technique used in Operations Research (namely, Lagrangian Relaxation), it has a clear logical

semantics, that smoothly integrates into the constraint model.

We implemented various general-purpose search-strategies, then we moved to a new search strategy that is more tailored to the given problem, obtaining strong improvements.

Experimental results show a large improvement with respect to the currently used solutions, showing that Logic Programming can be effective to address real life problems.

The logic program consists of about 300 lines of ECL^{i}PS^e code, including custom constraints, and the interfacing to the Java TSP solver, plus about 600 lines of Java code. The main objective of the application was to provide to nurses more balanced workloads, and to patients more continuity (loyalty) in the service. In other words, the objective was to improve the feeling of the quality of working conditions for the nurses and the perception of the quality of service for the patients. However, as a by-product, we also reduced the workload of the nurses, in terms of travel time. With respect to the hand-made solution, a nurse saves about 3 hours per week. In this way, the working time of the nurses is used more effectively to provide service to patients, instead of travelling on sub-optimal routes. The saved time could be used to provide better service to the patients, or to serve more patients, which is a strong improvement in a period of crisis and governmental cuts.

The application was mainly designed, developed and tested by two PhD students in about six months. As a rough estimate, the person-months for the development will be returned in about 8 months, which shows that Logic Programming can be an economically affordable technology to improve working conditions and service quality.

References

1 David L. Applegate, Robert E. Bixby, Vasek Chvátal, and William J. Cook. *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, 2006.

2 K. Apt and M. Wallace. *Constraint logic programming using ECL^{i}PS^e*. 2007.

3 Thomas Bartz-Beielstein, Marco Chiarandini, Luís Paquete, and Mike Preuss, editors. *Experimental Methods for the Analysis of Optimization Algorithms*. Springer, Germany, 2010.

4 N. Beldiceanu and E. Contejean. Introducing global constraints in CHIP. *Mathematical and Computer Modelling*, 20(12):97 – 123, 1994.

5 S. Bertels and T. Fahle. A hybrid setup for a hybrid scenario: combining heuristics for the home health care problem. *Computers & OR*, 33(10), 2006.

6 Yves Caseau and François Laburthe. Solving small TSPs with constraints. In L. Naish, editor, *ICLP*. The MIT Press, 1997.

7 P. Eveborn, M. Rönnqvist, H. Einarsdóttir, M. Eklund, K. Lidén, and M. Almroth. Operations research improves quality and efficiency in home care. *Interfaces*, 2009.

8 Filippo Focacci, Michela Milano, and Andrea Lodi. Soving TSP with time windows with constraints. In Danny De Schreye, editor, *ICLP*, pages 515–529. MIT Press, 1999.

9 R. Herrero, J.J. Ramos, and D. Guimarans. Lagrangian metaheuristic for the travelling salesman problem. In *Extended abstracts of Operational Research Conference 52*, Royal Holloway, University of London, September 2010.

10 L. Kaya and J. Hooker. A filter for the circuit constraint. In F. Benhamou, editor, *CP*, volume 4204 of *Lecture Notes in Computer Science*, pages 706–710. Springer, 2006.

11 M. Luby, A. Sinclair, and D. Zuckerman. Optimal speedup of Las Vegas algorithms. *Inf. Process. Lett.*, 1993.

12 G. Pesant, M. Gendreau, J-Y. Potvin, and J-M. Rousseau. An exact constraint logic programming algorithm for the TSP with time windows. *Transp. Science*, 32(1), 1998.

13 J.-C. Régin. A filtering algorithm for constraints of difference in CSPs. In B. Hayes-Roth and R. Korf, editors, *AAAI*, pages 362–367. AAAI Press / The MIT Press, 1994.

14 H. Simonis. Models for global constraint applications. *Constraints*, 12:63–92, 2007.