Solving Nurse Rostering Problems Using Soft Global Constraints

Jean-Philippe Metivier, Patrice Boizumault, Samir Loudni

Solving Nurse Rostering Problems Using Soft Global Constraints

Jean-Philippe Métivier, Patrice Boizumault, and Samir Loudni

GREYC (CNRS - UMR 6072) – Université de Caen
Campus II – Boulevard du Maréchal Juin
14000 Caen Cedex

Abstract. Nurse Rostering Problems (NRPs) consist of generating rosters where required shifts are assigned to nurses over a scheduling period satisfying a number of constraints. Most NRPs in real world are NP-hard and are particularly challenging as a large set of different constraints and specific nurse preferences need to be satisfied. The aim of this paper is to show how NRPs can be easily modelled and efficiently solved using soft global constraints. Experiments on real-life problems and comparison with ad'hoc OR approaches are detailed.

1 Introduction

Due to their complexity and importance in real world modern hospitals, Nurse Rostering Problems (NRPs) have been extensively studied in both Operational Research (OR) and Artificial Intelligence (AI) for more than 40 years [5,11]. Most NRPs in real world are NP-hard [16] and are particularly challenging as a large set of different rules and specific nurse preferences need to be satisfied to warrant high quality rosters for nurses in practice. Other wide range of heterogeneous and specific constraints makes the problem over-constrained and hard to solve efficiently [1,27].

NRPs consist of generating rosters where required shifts are assigned to nurses over a scheduling period satisfying a number of constraints [5,7]. These constraints are usually defined by regulations, working practices and preferences of nurses and are usually categorised into two groups: hard constraints and soft constraints (with their violation costs).

From a Constraint Programming (CP) point of view, global constraints are often key elements in successfully modelling and solving real-life problems due to their efficient filtering. Global constraints are particularly well suited [31] for modelling NRPs: sequence constraints on every nurse planning, daily capacity constraints, etc. But, for over-constrained problems as NRPs, such filtering can only be performed in very particular cases. Soft global constraints proposed by [30,26,33] take advantage from the semantics of a constraint and from the semantics of its violation to efficiently perform filtering. The aim of this paper is to show how NRPs can be modelled using soft global constraints and solved efficiently with solutions quality and computing times close to those obtained using ad'hoc OR methods.

Section 2 gives a synthetic overview of NRPs and describes the problem we selected as example for our presentation. Although this problem is fictional, hard and soft constraints it contains are representative of constraints encountered in most NRPs. Section 3 is devoted to soft global constraints and their filtering. In Section 4, we show how NRPs can be modelled in a concise and elegant way using soft global constraints.

The next two sections are devoted to the resolution of NRPs. First, we introduce (Section 5) the global constraint `regularCount` (and its soft version) which combines `regular` and `atleast`/`atmost` constraints in order to provide a more efficient filtering. Then, we motivate and present our resolution method VNS/LDS+CP [18] based on a Variable Neighborhood Search (VNS [22]) where the reconstruction step is performed using a Limited Discrepancy Search (LDS [13]) combined with filtering (CP) performed by soft global constraints.

The ASAP site (Automated Scheduling, optimization And Planning) of University of Nottingham (`http://www.cs.nott.ac.uk/~tec/NRP/`) records a large and various set of NRPs instances as well as the methods used to solve them. We performed experimentations over different instances we selected in order to be representative of the diversity and the size of NRPs. For each instance, we compare quality of solutions and computing times for our method with the best known method for solving it [14]. Experimentations show (Section 7) that, despite its genericity and flexibility, our method provides excellent results on small and middle size problems and very promizing results on large scale problems.

2 Nurse Rostering Problems

2.1 An Overview of NRPs

NRPs consist of generating rosters where required *shifts* are assigned to nurses over a scheduling period (*planning horizon*) satisfying a number of constraints [5,11]. These constraints are usually defined by regulations, working practices and nurses preference. Constraints are usually categorised into two groups: *hard* and *soft* ones. Hard constraints must be satisfied in order to obtain feasible solutions for use in practice. A common hard constraint is to assign all shifts required to the limited number of nurses. Soft constraints are not obligatory but are desired to be satisfied as much as possible. The violations of soft constraints in the roster are used to evaluate the quality of solutions. A common soft constraint in NRPs is to generate rosters with a balanced workload so that human resources are used efficiently.

Shift types are hospital duties which usually have a well-defined start and end time. Many nurse rostering problems are concerned with the three traditional shifts Morning, (7:00–15:00), Evening (15:00–23:00), and Night (23:00–7:00). *Shift constraints* express the number of personnel needed for every skill category and for every shift or time interval during the entire planning period. *Nurse constraints* refer to all the restrictions on personal schedules. All the personal requests, personal preferences, and constraints on balancing the workload among personnel belong to this category.

2.2 Example: A 3-Shifts NRP

The 3 shifts are Morning (M), Evening (E) and Night (N). Off (O) will represent repose. 8 nurses must be planned over a planning horizon of 28 days satisfying the below constraints.

1. **Hard Constraints:**

   (H1) From Monday to Friday, M, E, N shifts require respectively (2, 2,1) nurses. For weekend, the demand of all shifts is reduced to 1.
   (H2) A nurse must have at least 10 days off.
   (H3) A nurse must have 2 free Sundays.
   (H4) A nurse is not allowed to work more than 4 N shifts.
   (H5) The number of consecutive N shifts is at least 2 and at most 3.
   (H6) Shift changes must be performed respecting the order: M, E, N.

2. **Soft Constraints:**

   (S1) For a nurse, the number of M and N shifts should be within the range [5..10]. Any deviation δ is penalised by a cost $\delta \times 10$.
   (S2) The number of consecutive working days is at most 4. Any excess δ generates a penality of $\delta \times 1000$.
   (S3) Every isolated day off is penalised by a cost 100.
   (S4) Every isolated working day is penalised by a cost 100.
   (S5) Two working days must be separated by 16 hours of rest. Any violation generates a cost 100.
   (S6) An incomplete weekend has cost 200.
   (S7) Over a period of 2 weeks a nurse must have 2 days off during weekends. Any deviation δ is penalised by a cost $\delta \times 100$.
   (S8) A N shift on Friday before a free weekend is penalised by a cost 500.

Related CP Works. An operational system GYMNASTE using (hard) global constraints is described in [31]. Other practical systems are also mentionned. But, dealing with over-constrained problems is only discussed as perspectives. Three directions are indicated (Hierarchical CP [1], heuristics and interactions). The last two proposals are problem dependent. As quoted by [31], the main difficulty with the Hierarchical CP approach is that global constraints have to be extended to handle constraint violations. This is the aim of this paper.

3 Soft Global Constraints

3.1 Principles

Over-constrained problems are generally modelled as Constraint Optimization Problems (COP). A cost is associated to each constraint in order to quantify its violation. A global objective related to the whole set of costs is usually defined (for example to minimize the total sum of costs). Global constraints are often key elements in successfully modelling and solving real-life problems due to their efficient filtering. But, for over-constrained problems, such filtering can only be performed in very particular cases. Soft global constraints proposed by [30,26,33] take advantage from the semantics of a constraint and from the semantics of its violation to efficiently perform filtering.

Definition 1 (violation measure). *μ is a violation measure for the global constraint* $\texttt{c}(X_1, ..., X_n)$ *iff μ is a function from* $D_1 \times D_2 \times ... \times D_n$ *to* $\Re^+$ *s.t.* $\forall A \in D_1 \times D_2 \times ... \times D_n$, $\mu(A) = 0$ *iff A satisfies* $c(X_1, ..., X_n)$.

To each soft global constraint c are associated a violation measure μ_c and a cost variable Z_c that measures the violation of c. So the COP is transformed into a CSP where all constraints are hard and the cost variable $Z = \sum_c Z_c$ will be minimized. If the domain of a cost variable is reduced during the search, propagation will be performed on domains of other cost variables.

Definition 2 (soft global constraint). *Let* $\texttt{c}(X_1, ..., X_n)$ *be a global constraint, Z_c its cost variable, and μ a violation measure. The soft global constraint* $\Sigma\texttt{-c}([X_1, ..., X_n], \mu, Z_c)$ *has a solution iff* $\exists A \in D_1 \times D_2 \times ... \times D_n$ *s.t.* $min(D_{Z_c}) \leq \mu(A) \leq max(D_{Z_c})$.

Example (decomposition based violation measure). Let $\texttt{c}(X_1, ..., X_n)$ be a global constraint which can de decomposed as a conjunction of binary constraints $c_{i,j}$ over variables X_i and X_j. Let $\varphi_{i,j}$ be the violation cost of $c_{i,j}$; let $A \in D_1 \times D_2 \times ... \times D_n$ and $unsat(A)$ be the set of constraints $c_{i,j}$ unsatisfied by A. Then, $\mu_{dec}(A) = \sum_{c_{i,j} \in unsat(A)} \varphi_{i,j}$.

3.2 Relaxation of gcc

i) A Global Cardinality Constraint (gcc) on a sequence of variables specifies, for each value in the union of their domains, an upper and lower bound to the number of variables that are assigned to this value [28].

Definition 3. *Let* $\mathcal{X}$=$\{X_1, \ldots, X_n\}$, $Doms$=$\cup_{X_i \in \mathcal{X}} D_i$. *Let* $v_j \in Doms$, l_j *and* u_j *the lower and upper bounds for* v_j. $\texttt{gcc}(\mathcal{X}, l, u)$ *has a solution iff* $\exists A \in D_1 \times D_2 \times ... \times D_n$ *s.t.* $\forall v_j \in Doms$, $l_j \leq |\ \{X_i \in \mathcal{X} \mid X_i = v_j\}\ | \leq u_j$.

Each constraint $\texttt{gcc}(\mathcal{X}, l, u)$ can be decomposed as a conjunction of `atleast` and `atmost` constraints over values in $Doms$:

$$\texttt{gcc}(\mathcal{X}, l, u) = \bigwedge_{v_j \in Doms} (\texttt{atleast}(\mathcal{X}, v_j, l_j) \wedge \texttt{atmost}(\mathcal{X}, v_j, u_j))$$

ii) Global constraint Σ-gcc is a soft version of gcc for the decomposition based violation measure μ_{dec}[20]. Σ-gcc allows the violation of the lower and/or upper bounds of values. To each value $v_j \in Doms$ are associated a *shortage* function $s(\mathcal{X}, v_j)$ measuring the number of missing assignments of v_j to satisfy $\texttt{atleast}(\mathcal{X}, v_j, l_j)$, and an *excess* function $e(\mathcal{X}, v_j)$ measuring the number of assignments of v_j in excess to satisfy $\texttt{atmost}(\mathcal{X}, v_j, u_j)$. Each constraint Σ-gcc is modelled by adding *violation arcs* to the network of gcc [28]. These violation arcs represent the shortage or the excess for each value of $Doms$ [33,20].

Definition 4 (μ_{dec}). *For each value* $v_j \in Doms$, *let* $\varphi_j^{atleast}$ *be the violation cost of its lower bound* l_j *and* φ_j^{atmost} *the violation cost of its upper bound* u_j,

$$\mu_{dec}(\mathcal{X}) = \sum_{v_j \in Doms} \mu_{card}(\mathcal{X}, v_j)$$
$$where\, \mu_{card}(\mathcal{X}, v_j) = s(\mathcal{X}, v_j) \times \varphi_j^{atleast} + e(\mathcal{X}, v_j) \times \varphi_j^{atmost}$$

*Property 1 (***filtering).** Let Z be a cost variable and Φ a violation structure, Σ-gcc$(\mathcal{X}, l, u, \mu_{dec}, \Phi, Z)$ is domain-consistent iff for every arc $a = (X_i, v_j)$, there exists an integer $s - t$ flow f of value $max(n, \sum_{v_j \in Doms} l_j)$ with $f(a) = 1$ and weight p s.t. $min(D_Z) \leq p \leq max(D_Z)$.

Worst case time complexity: $O(n^2 log(n) \times d)$ where $d = max(|D_i|)$.

3.3 Relaxation of `Regular`

i) Global Constraint `Regular` [25]

Definition 5. *Let M be a Deterministic Finite Automaton (DFA), $\mathcal{L}(M)$ the language defined by M, $\mathcal{X}$ a sequence of n variables.* `regular`*$(\mathcal{X}$,$M)$ has a solution iff $\exists A \in D_1 \times D_2 \times ... \times D_n$ s.t. $A \in \mathcal{L}(M)$.*

A DFA is defined by a 5-tuple $M = \{Q, \Sigma, \delta, q_0, F\}$ where Q is a finite set of states, Σ is an alphabet, $\delta : Q \times \Sigma \rightarrow Q$ is a transition function, q_0 is the initial state and $F \subseteq Q$ is the set of final (or accepting) states. A `regular` constraint over a sequence of n variables is modelled by a layered directed graph $\mathcal{G} = (V, U)$:

- vertex set $V = \{s\} \cup V_0 \cup \ldots \cup V_n \cup \{t\}$ where $\forall i \in [1..n], V_i = \{q_l^i \mid q_l \in Q\}$
- arc set $U = \{(s, q_0^0)\} \cup U_0 \cup \ldots U_n \cup \{(q_l^n, t) \mid q_l \in F\}$
  where $\forall i \in [1..n], U_i = \{(q_l^i, q_m^{i+1}, v_j) \mid v_j \in D_i, \delta(q_l, v_j) = q_m\}$

Property 2. Solutions for `regular`$(\mathcal{X}$,$M)$ correspond to s-t paths in graph $\mathcal{G}$.

ii) Global Constraint `Cost-Regular` enables to model the fact that some transitions of an automaton may have a cost. To each `cost-regular` constraint is associated a directed weighted layered graph $\mathcal{G} = (V, U, \Phi)$, where each arc representing a transition is valued by the cost of this transition [10]. For an instantiation A, the measure $\mu_{reg}(A)$ is defined as the total sum of the transition costs for arcs belonging to the path associated to A.

*Property 3 (***filtering).** Let Z be a cost variable and Φ a violation structure. `cost-regular`$(\mathcal{X}, M, \mu_{reg}, \Phi, Z)$ is domain-consistent iff for every arc $a = (X_i, v_j)$ there exists an s-t path of weight p s.t. $min(D_Z) \leq p \leq max(D_Z)$.

Worst case time complexity: $O(n \times |Q| \times |\Sigma|)$. For each layer i, each vertex q_l^i may have at most $|\Sigma|$ successors.

iii) `Cost-Regular` **used as a soft constraint for NRPs** every hard constraint of sequence c will be modelled using a DFA M_c (see Section 4.3). A soft constraint of sequence Σ-`c` will be modelled by adding new transitions to M_c as well as their costs. Let M_c' be this new DFA; then a `cost-regular` constraint over M_c' is stated. So, for modelling NRPs, `cost-regular` will be used as a soft version of `regular` ($\mathcal{L}(M_c) \subset \mathcal{L}(M_c')$).

4 Modelling a 3-Shifts NRP

This section presents the modelling of the problem specified in Section 2.2 and shows how soft global constraints are well suited for modelling NRPs.

4.1 Variables and Domains

Let J=[1..28] be the scheduling period and I=[1..8] the set of nurses; variable X_j^i, with domain D_j^i=Doms=$\{M,E,N,O\}$, will represent the shift assigned to nurse i for day j. For `gcc` constraints, values will be ordered as in $Doms$.

4.2 Capacity Constraints

(H1) $\forall j \in [1,2,3,4,5,8,9,...,24,25,26]$, $\texttt{gcc}([X_j^1,\ldots,X_j^8],[2,2,1,0],[2,2,1,8])$.
$\forall j \in [6,7,13,14,20,21,27,28]$, $\texttt{gcc}([X_j^1,\ldots,X_j^8],[1,1,1,0],[1,1,1,8])$.

(H2) $\forall i \in I, \texttt{atleast}([X_1^i,\ldots,X_{28}^i],O,10)$.

(H3) $\forall i \in I, \texttt{atleast}([X_7^i,X_{14}^i,X_{21}^i,X_{28}^i],O,2)$.

(H4) $\forall i \in I, \texttt{atmost}([X_1^i,\ldots,X_{28}^i],N,4)$.

(S1) $\forall i \in I, \Sigma\texttt{-gcc}([X_1^i,\ldots,X_{28}^i],[5,5],[10,10],[10,10],[10,10],Z_i)$ for values M and E with $\varphi_M^{atleast}$=φ_M^{atmost}=10 (violation costs for E are the same).

(S7) is also modelled using Σ-`gcc` constraints.

(H2), (H4) and (S1) can be grouped together using Σ-`gcc` constraints:

$\forall i \in I$, $\Sigma\texttt{-gcc}([X_1^i,\ldots,X_{28}^i],[5,5,0,10],[10,10,4,28],[10,10,0,\infty],[10,10,\infty,0],Z_i)$.
As (H2) is a hard `atleast` constraint, then u_O=28, $\varphi_O^{atleast}$=∞ and φ_O^{atmost}=0.
As (H4) is a hard `atmost` constraint, then l_N=0, $\varphi_N^{atleast}$=0 and φ_N^{atmost}=∞.

4.3 Sequence Constraints

Let Σ be an alphabet, $\overline{x}$ will represent any symbol $y \in \Sigma$ s.t. $y \neq x$.

(H5) $\forall i \in I$, $\texttt{regular}([X_1^i,\ldots,X_{28}^i],A_1)$ (Figure 1)

(H6) states that shift changes must be performed respecting the order: M, E, N. (see automaton A_2 (Figure 2)). For modelling (S5), two arcs are added: one for transition (e_3,e_2,E) with cost 100 and one for transition (e_2,e_1,M) with cost 100. So, $\forall i \in I$, $\texttt{cost-regular}([X_1^i,\ldots,X_{28}^i],A_2,Z_i^{A2})$.

(S6) $\forall\ i \in I$, $\texttt{cost-regular}([X_6^i,X_7^i,X_{13}^i,X_{14}^i,...,X_{27}^i,X_{28}^i,],A_3,Z_i^{A3})$ (Figure 3).

Finally, (S2), (S3) and (S4) can be grouped together using `cost-regular`. (S8) can also be modelled by `cost-regular`.

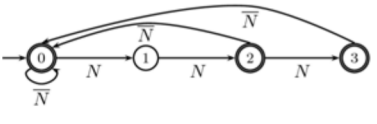

Fig. 1. Automaton A1 for (H5)

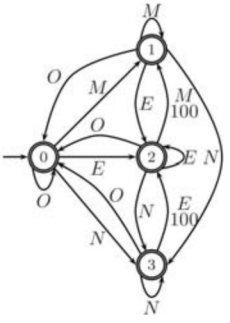

Fig. 2. Automaton A2 for (H6) and (S5)

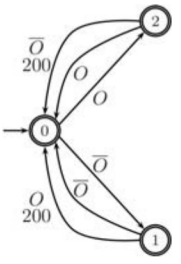

Fig. 3. Automaton A3 for (S6)

5 Interaction between Global Constraints

Despite the efficient filtering of each global constraint, the lack of communication between them reduces significantly the quality of the whole filtering. Indeed, each global constraint uses its internal representation (bipartite graph, network,...) and does not (or partially) exploit information deduced by other global constraints. In most NRPs, a great number of global constraints share a common set of variables (e.g. constraints over the entire planning of a nurse). Few works have been done on the interaction between global constraints:

- `cardinality matrix constraint` [29] combining several `gcc` as a matrix,
- `multi-cost-regular` [19] merging multiple `cost-regular`.

In this section, we propose the `regularCount` (resp. `cost-regularCount`) constraint which combines a `regular` (resp. `cost-regular`) constraint with several `atleast`/`atmost` constraints on a same value.

5.1 Motivating Example

Rule (H4) is modelled as: $\forall i \in I$, `atmost`($[X_1^i,\ldots, X_7^i]$,N,4) and rule (H5) as: $\forall i \in I$, `regular`($[X_1^i,\ldots, X_7^i]$,A_1) (see Section 2.2 & Section 4.3). Let us consider the following reduced variable domains associated to the first week of nurse i: $D_1^i{=}D_2^i{=}D_3^i{=}\{N\}$, $D_4^i{=}\{O\}$ and $D_5^i{=}D_6^i{=}D_7^i{=}\{N,O\}$. Filtering separately `atmost` and `regular` will not detect that value N should be removed from D_5^i. Indeed, if $X_5^i{=}N$ then $X_6^i{=}N$ by (H5) but (H4) fails. For analogous reasons, value N should also be removed from D_6^i and D_7^i. This example illustrates the weakness of separate filterings.

5.2 `regularCount` Constraint

For an automaton and a particular value $v_j \in Doms$, a `regularCount` constraint will combine a `regular` constraint with several `atleast`/`atmost` constraints on value v_j.

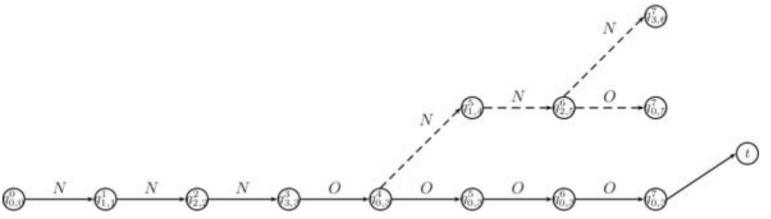

Fig. 4. Graph representation for `regularCount`([X_1^i,..., X_7^i],A_1,N,0,4)

Definition 6. *Let M=$\{Q, \Sigma, \delta, q_0, F\}$ be a DFA, $\mathcal{L}(M)$ its associated language, $\mathcal{X}$ a sequence of n variables, $v_j \in \Sigma$, l_j (resp. u_j) an upper (resp. lower) bound for v_j.* `regularCount`*($\mathcal{X}$,M,v_j,l_j,u_j) has a solution iff* $\exists A \in D_1 \times \ldots \times D_n \mid A \in \mathcal{L}(M) \wedge l_j \leq \mid \{X_i \in \mathcal{X} \mid X_i = v_j\} \mid \leq u_j$.

The `regularCount` constraint has been used for developping a track planner for a local Radio station. A new track has to be broadcast a bounded number of times into the daily planning which must respect musical transitions expressed as regular expressions. Other potential use for `regularCount` would be planning advertising for TV stations or planning maintenance periods for assembly-lines.

Graph Representation. As for `regular` (see Section 3.3), a constraint `regularCount` ($\mathcal{X}$,M,v_j,l_j,u_j) is modelled by a layered directed graph $\mathcal{G}'(V, U)$. For each layer i, states q_l are labelled both by the layer (q_l^i as for `regular`) and by k the number of occurrences of v_j found so far ($q_{l,k}^i$ for `atleast` and `atmost` constraints).

$$
\begin{array}{l}
V = \{s\} \cup V_0 \cup \ldots \cup V_n \cup \{t\} \\
V_i = \{q_{l,k}^i \mid q_l \in Q, k \in [0 \ldots n]\},\ \forall i \in [1..n] \\
U = \{(s, q_{0,0}^0)\} \cup U_0 \cup \ldots U_n \cup U_t \\
U_t = \{(q_{l,k}^n, t) \mid q_l \in F \wedge (l_j \leq k \leq u_j)\} \\
U_i = \{(q_{l,k}^i, q_{m,k'}^{i+1}, v) \mid v \in D_i, \delta(q_l, v) = q_m\},\ \forall i \in [1..n] \\
\textit{where}\ \ \text{if}\,(v = v_j)\,\text{then}\,k' = k + 1\,\text{else}\,k' = k
\end{array}
$$

Example (Section 5.1) is modelled as: $\forall i \in I$, `regularCount`([X_1^i,..., X_7^i],A_1,N,0,4). Figure 4 describes its graph representation. `regularCount` filtering will remove value N from domains D_5^i, D_6^i and D_7^i.

Property 4 **(filtering).** Let M=$\{Q, \Sigma, \delta, q_0, F\}$ be a DFA, $v_j \in \Sigma$, l_j (resp. u_j) the upper (resp. lower) bound for v_j. `regularCount`($\mathcal{X}$,M,v_j,l_j,u_j) is domain-consistent iff for every arc $a = (X_i, v_j) \in U_i$ there exists an s-t path in $\mathcal{G}'(V, U)$.

Proof. There is an arc from $q_{l,k}^i$ to $q_{m,k'}^{i+1}$ iff there exists a value $v \in D_i$ such that $\delta(q_l, v) = q_m$. If $(v = v_j)$ then $k' = k + 1$ (i.e., the number of variables that are assigned to v_j is $k + 1$), otherwise k'=k. If an arc belongs to an s-t path, it belongs to a path from $q_{0,0}^0$ to $q_{l,k}^n$, with $q_l \in F$ and $l_j \leq k' \leq u_j$.

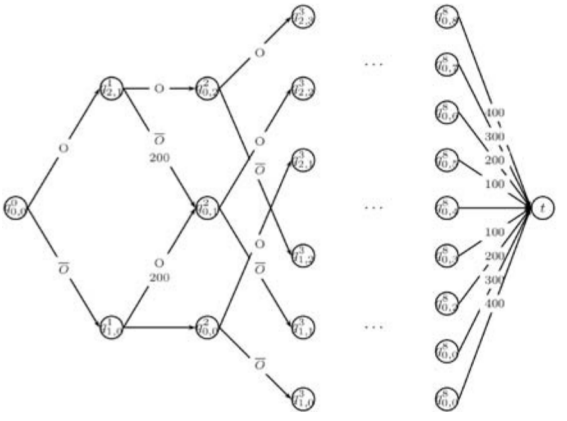

Fig. 5. Graph representation for `cost-regularCount`($[X_6^i,...,X_{28}^i]$,A_3,Φ,O,2,2,$Z_i^{A_3}$)

Worst case time complexity: $O(n^2/2 \times |Q| \times |\Sigma|)$. For layer i, there are at most $i \times |Q|$ vertices as at most i occurrences of v_j may be assigned to variables $X_1, ..., X_i$. As for `regular`, each vertex may have at most $|\Sigma|$ successors. Summing for all layers leads to $\sum_{i=1}^{n} i \times |Q| \times |\Sigma|$.

5.3 `Cost-regularCount` Constraint

Let $M=\{Q, \Sigma, \delta, q_0, F\}$ be a DFA and $v_j \in \Sigma$. `cost-regularCount` is a soft version of `regularCount` for which some transitions may have a cost and which also allows the violation of lower/upper bounds for a value v_j.

Definition 7 (violation measure $\overline{\mu}_{reg}$). *Let $\varphi_j^{atleast}$ (resp. φ_j^{atmost}) be the violation cost associated to lower bound l_j (resp. upper bound u_j). $\forall v_j \in \Sigma$, $\overline{\mu}_{reg}(\mathcal{X}, v_j) = \mu_{reg}(\mathcal{X}) + \mu_{card}(\mathcal{X}, v_j)$.*

Definition 8. *Let M a DFA, $v_j \in \Sigma$, Z a cost variable and Φ a violation structure.* `cost-regularCount`*$(\mathcal{X}, M, \overline{\mu}_{reg}, \Phi, v_j, l_j, u_j, Z)$ has a solution iff $\exists A \in D_1 \times \ldots \times D_n$ s.t. $A \in \mathcal{L}(M) \wedge \min(D_Z) \leq \overline{\mu}_{reg}(A, v_j) \leq \max(D_Z)$.*

Graph Representation. There are two main differences between $\mathcal{G}''(V, U, \Phi)$ and $\mathcal{G}'(V, U)$: i) transition costs are associated to corresponding arcs (as for `cost-regular`), ii) arcs U_t are replaced by violation arcs $\tilde{U}_t = \{(q_{l,k}^n, t) \mid q_l \in F, k \in [0 \ldots n]\}$ which enable to model shortage or excess for a value v_j. To each violation arc $a=(q_{l,k}^n, t)$ is associated a cost $w(a)$:

$$w(a) = \begin{cases} (l_j - k) \times \varphi_j^{atleast} & \text{if } k < l_j \\ (k - u_j) \times \varphi_j^{atmost} & \text{if } k > u_j \\ 0 & \text{otherwise} \end{cases}$$

Figure 5 gives the graph representation of (S6) and (S7) modelled as:

$\forall i \in I$, `cost-regularCount`$([X_6^i, X_7^i, X_{13}^i, ..., X_{27}^i, X_{28}^i,], A_3, \overline{\mu}_{reg}, \Phi, O, 2, 2, Z_i^{A_3})$.

Table 1. Comparative results for Filtering. ($\star$) denotes optimal values.

Instance	$\|I\| \times \|J\|$	$\|D\|$	UB	Σ-Gcc & cost-regular		cost-regularCount	
				Time (s.)	♯backtracks	Time (s.)	♯backtracks
inst_01_07	28	2	3000*	1.4	1 559	**0.2**	**342**
inst_01_11	44	2	1500*	20.9	14 113	**6.7**	**6 002**
inst_01_14	56	2	2500*	380.1	193 156	**122.6**	**63 395**
inst_02_07	49	3	1100*	$\geq$5 400	–	**3 303.1**	**2 891 874**
inst_02_14	98	3	100*	**73.6**	24 100	120.2	**16 587**
inst_02_21	147	3	100*	4 886.7	1 216 908	**940.5**	**107 612**

*Property 5 (***filtering).** Let $M=\{Q, \Sigma, \delta, q_0, F\}$ be a DFA, $v_j \in \Sigma$, l_j (resp. u_j) the upper (resp. lower) bound for v_j, Φ a violation structure and Z a cost-variable. cost-regularCount$(\mathcal{X}, M, \overline{\mu}_{reg}, \Phi, v_j, l_j, u_j, Z)$ is domain consistent iff for every arc $a = (X_i, v) \in U_i$ there exists an s-t path in $\mathcal{G}''(V, , \Phi)$ of cost p s.t. $\min(D_z) \leq p \leq \max(D_z)$.

Proof. If a transition associated to arc $(q^i_{l,k}, q^{i+1}_{m,k'})$ uses v_j, then $k'=k+1$ else $k'=k$. An s-t path using a violation arc in $\tilde{U}_t$ corresponds to a solution with a shortage or an excess for value v_j and the cost to pay is reported on this violation arc. As transition costs are reported on their associated arcs in U_i, the cost of an s-t path A using a violation arc in $\tilde{U}_t$ corresponds exactly to $\overline{\mu}_{reg}(A, v_j)$.

Worst case time complexity: $O(n^2/2 \times |Q| \times |\Sigma|)$ as for regularCount.

Table 1 compares the efficiency of cost-regularCount filtering vs separate filterings in terms of computing times and number of backtracks. Experiments have been performed on small or medium NRPs instances using Depth First Branch and Bound and run on a 2.8 Ghz P4 processor. cost-regularCount filtering always performs better than separate filterings except for inst_02_14 which is an instance easy to solve where filtering is not so crucial. The extra-cost comes from the fact that the complexity of cost-regularCount is slightly higher than those of cost-regular and Σ-gcc.

6 Variable Neighborhood Search

A great variety of approaches that have been proposed for solving NRPs are either ad'hoc OR methods (including preprocessing steps to reduce the problem size), or local search methods combining OR techniques to find an initial solution. NRPs seem to be well suited for defining large-scale neighborhoods (2-opt, swap and interchange of large portions of nurse plannings, ...).

Variable Neighborhood Search (VNS) [22] is a metaheuristic which systematically exploits the idea of large neighborhood change, both in descent to local minima and in escape from the valleys which contain them. Variable Neighborhood Decomposition Search (VNDS) [12] extends basic VNS within a successive

Algorithm 1. Pseudo-code for VNS/LDS+CP.

```
function VNS/LDS+CP(X, C, k_init, k_max, δ_max)
begin
    s ← genInitialSol(X)
    k ← k_init
    while (k < k_max) ∧ (not timeout) do
        X_unaffected ← H_neighbor(N_k, s)
        A ← s\{(x_i = a) s.t. x_i ∈ X_unaffected}
        s' ← NaryLDS(A, X_unaffected, δ_max, V(s), s)
        if V(s') < V(s) then
            s ← s'
            k ← k_init
        else k ← k + 1
    return s
end
```

approximations method. For a solution of size n, all but k variables are fixed, and VNDS solves a sub-problem in the space of the k unfixed variables.

VNS/LDS+CP. [18] is a generic local search method based on VNDS [12]. Neighborhoods are obtained by unfixing a part of the current solution according to a neighborhood heuristic. Then the exploration of the search space related to the unfixed part of the current solution is performed by a partial tree search (LDS, [13]) with CP in order to benefit from the efficiency of global constraints filtering (See Algorithm 1). However, as the size of neighborhoods can quickly grow, the exploration of (very) larger neighborhoods may require a too expensive effort. That is why, in order to efficiently explore parts of the search space, LDS is used.

LDS+CP: Our *variable ordering* for LDS is `Dom/Deg` and our *value ordering* selects the value which leads to the lowest increase of the violation cost. CP is performed using soft global constraints filtering.

Neighborhood Heuristics. A lot of soft global constraints are stated over the whole planning of a nurse. So, all variables related to a nurse planning will be together unassigned. k will represent the number of nurse plannings to be unassigned (and not the number of variables as depicted in general Algorithm 1). In order to show the interest of soft global constraints filtering, only two "basic" heuristics have been considered: (i) `rand` which randomly selects a nurse planning, and (ii) `maxV` which selects the nurse planning having the highest violation cost.

7 Experimental Results

We performed experimentations over different instances we selected in order to be representative of the diversity and the size of NRPs. For each instance, **we always compare with the best method** for solving it [14]. As experiments

have been run on various machines, we will report, for each instance, the original CPU time and the processor. For all instances, except the first three ones where the processor is unknown (they are noted in italic Table 1.), CPU times will be normalised[1] and denoted `CPUN`. **Some methods include a pre-treatment.** As CPU times for this step are not given in papers, reported CPU times concern in fact the second step. Finally, reported CPU times for our method always **include the computing time for obtaining the initial solution**.

Experimental Protocol. Each instance has been solved by VNS/LDS+CP, with a discrepancy varying from δ_{min}=3 to δ_{max}=8. k_{min} has been set to 2 and k_{max} to 66% of the total number of nurses. Timeout has been set according to the size of each instance. For the `rand` heuristic, a set of 10 runs per instance has been performed. VNS/LDS+CP has been implemented in C++. Experiments have been performed under Linux on a 2.8 Ghz P4 processor, with 1GB RAM.

Comparisons with ad'hoc Methods. Heuristic `maxV`, which is better informed that `rand`, provides the best performances except for instances `LLR` and `Azaiez`.

A) `Ozkarahan` *instance [24]: we find the optimum in less than* 1*s. using* `maxV`.

B) `MILLAR` *instance:* (2 methods)
B1) Network programming [21]: All feasible weekly shift patterns of length at most 4 days are generated. An ILP model is defined and solved using CPLEX.
B2) TS+B&B [15]: Nurse constraints are used to produce all feasible shift patterns for the whole scheduling period for each nurse (independently from shift constraints). Best combinations of these shift patterns are found using mathematical programming and Tabu Search.

With B1, a solution of cost 2.550 is found after 500 s. on an IBM RISC6000/340. With B2, a solution of cost 0 is obtained in 1 s. on a 1Ghz Intel P3 processor. *We find the optimum in less than* 1 *s. using* `maxV`.

C) `Musa` *instance [23]:* A solution of cost 199 is found in 28 s. on UNIVAC-1100. *We find the optimum (cost 175) in* 39 *s. using* `maxV`.

D) `LLR` *instance:* A hybrid AI approach (TS+CP), which combines CP techniques and Tabu Search is used in [17]. A solution of cost 366 is found after 96 s. on a PC/P-545MHz (`CPUN` 16 s.). *With* `rand`*, we obtain (on average) a solution of cost* 319 *after* 265 *s. The best solution (over the 10 runs) has a cost* 314 *(*79 *s.). The first solution (cost* 363*) is obtained in less than* 1 *s. Using* `maxV`*, a solution of cost* 326 *is found in* 186 *s.*

E) `BCV-5.4.1` *instance:* (3 methods). All the results are obtained on a same machine (2.66GHz Intel P4 processor). Hybrid Tabu search [4] is the best of the 3 methods for this instance. VDS [6] finds the optimum after 135 s. (`CPUN` 128 s.). In [3], a solution of cost 200 is found after 16 s. (`CPUN` 15 s.). *With* `maxV`*, we obtain the optimum after* 180 *s.*

[1] For a machine κ times slower than ours, reported CPU times will be divided by κ.

Table 2. Comparative results. ($\star$) denotes optimal values.

Instances	$\|I\| \times \|J\|$	$\|D\|$	Best Ub	Ad'hoc methods			VNS/LDS+CP	
				Algo.	**Cost**	**Time(s)**	**Cost**	**Time(s)**
`Ozkarahan`	14×7	3	$0^\star$	[24]	-	-	**0**	1
`Millar`	8×14	3	$0^\star$	Network	2550	*500*	**0**	1
				TS+B&B	**0**	*1*		
`Musa`	11×14	2	$175^\star$	[23]	199	*28*	**175**	39
`LLR`	26×7	4	$301^\star$	TS+CP	366	16	314	79
`BCV-5.4.1`	4×28	5	$48^\star$	Hybrid TS	**48**	5	**48**	180
				VDS	**48**	128		
				[3]	200	15		
`Azaiez`	13×28	3	$0^\star$	(0,1)-LGP	**0**	150	**0**	233
`GPOST`	8×28	3	$3^\star$	2-Phase	**3**	14	8	234
`Valouxis`	16×28	4	$20^\star$	VDS	120	4200	160	3780
`Ikegami 3Shift-DATA1`	25×30	4	6	TS+B&B	6	111060	63	671

F) `Azaiez` *instance:* An optimal solution is provided with the (0,1)-LGP method [2] after 600 s. on a PC/P-700MHz (`CPUN` 150 s.). `rand` *(resp.* `maxV`*) finds the optimum in* 233 *s. (resp.* 1.050 *s.).*

G) `GPOST` *instance* is solved using a 2 steps method [14]. First, all feasible schedules are enumerated for each nurse. Then, the final schedule is generated using CPLEX. An optimal solution of cost 3 is found in 8 s. on a 2.83GHz Intel Core2 Duo processor (`CPUN` 14 s.) without taking into account the time used in the first step [14]. *We find a solution of cost* 8 *in* 234 *s. using* `maxV`*.*

H) `Valouxis` *instance [32]:* In [6], Variable Depth Search (VDS) obtains a solution of cost 120 (6 workstretches of length 3) after 2.993 s. on a 2.66GHz Intel Core2 Duo processor (`CPUN` 4.200 s.). *We obtain a solution of cost* 160 *(*8 *workstretches of length 3) after* 3.780 *s. using* `maxV`*.*

I) `Ikegami-3shift-DATA1` *instance:* Experiments have been performed on a PENTIUM3-1Ghz. TS+B&B [15] finds a solution of cost 10 after 543 mns (`CPUN` 194 mns) with a timeout of 24h and a solution of cost 6 after 5.783 mns (`CPUN` 1.851 mns) with a timeout of 100h. `maxV` *provides a solution of cost* 63 *(where all unsatisfied constraints are of weight* 1*) after* 671 *s. with a timeout of* 1*h.*

Contrary to other instances, nurse constraints are hard and shift constraints are soft for `Ikegami`. So our neighborhood heuristics which unassign whole nurse plannings are irrelevant. If the timeout is increased, the solution quality is improved but it is not enough to bring the optimum. As it is more efficient to unassign variables related to soft constraints than hard ones, one may consider that basic heuristics unassigning shift constraints would be efficient. But it is

not the case as it is very difficult to obtain a first solution: nurses constraints are larger than soft ones and more difficult to satisfy.

Conclusions. For each instance, we have compared our method with the best ad'hoc method for solving it [14]. Despite its genericity and flexibility, our method has obtained: (i) solutions of better quality and better computing times for `Ozkarahan`, `Millar`, `Musa` and `LLR`, (ii) solutions of equal quality with computing times close to those for `BCV541` and `Azaiez`, (iii) *very promising solution quality* on large scale instances as `GPOST` and `Valouxis`.

8 Conclusion

In this paper, we have shown how NRPs can be modelled in a concise and elegant way using soft global constraints. For each instance, we have compared quality of solutions and computing times for our method with the best known method for solving it. Experimentations show that, despite its genericity and flexibility, our method provides excellent results on small and middle size problems and very promizing results on large scale problems. For large instances or very specific ones like `Ikegami`, performances of our method could be greatly improved by using neighborhood heuristics especially designed for NRPs. In order to reduce the lack of communication between soft global constraints, it would be interesting to extend arc consistency for soft binary constraints [9,8].

References

1. Meyer auf'm Hofe, H.: Solving rostering tasks as constraint optimisation. In: Burke, E., Erben, W. (eds.) PATAT 2000. LNCS, vol. 2079, pp. 191–212. Springer, Heidelberg (2001)
2. Azaiez, M., Al Sharif, S.: A 0-1 goal programming model for nurse scheduling. Computers and Operations Research 32(3), 491–507 (2005)
3. Brucker, P., Burke, E., Curtois, T., Qu, R., Vanden Berghe, G.: A shift sequence based approach for nurse scheduling and a new benchmark dataset. J. of Heuristics (to appear, 2009)
4. Burke, E., De Causmaecker, P., Vanden Berghe, G.: A hybrid tabu search algorithm for the nurse rostering problem. In: McKay, B., Yao, X., Newton, C.S., Kim, J.-H., Furuhashi, T. (eds.) SEAL 1998. LNCS (LNAI), vol. 1585, pp. 187–194. Springer, Heidelberg (1999)
5. Burke, E., De Causmaecker, P., Vanden Berghe, G., Van Landeghem, H.: The state of the art of nurse rostering. Journal of Scheduling 7(6), 441–499 (2004)
6. Burke, E., Curtois, T., Qu, R., Vanden Berge, G.: A time predefined variable depth search for nurse rostering, TR 2007-6, University of Nottingham (2007)
7. Burke, E., Li, J., Qu, R.: A hybrid model of Integer Programming and VNS for highly-constrained nurse rostering problems. In: EJOR 2009 (to appear, 2009)
8. Cooper, M., de Givry, S., Sanchez, M., Schiex, T., Zytnicki, M.: Virtual arc consistency for Weighted CSP. In: AAAI 2008 (2008)
9. Cooper, M., Schiex, T.: Arc consistency for soft constraints. Artificial Intelligence 154(1-2), 199–227 (2004)

10. Demassey, S., Pesant, G., Rousseau, L.-M.: A `cost-regular` based hybrid column generation approach. Constraints 11(4), 315–333 (2006)
11. Ernst, A., Jiang, H., Krishnamoorthy, M., Sier, D.: Staff scheduling and rostering: A review of applications, methods and models. EJOR 153(1), 3–27 (2004)
12. Hansen, P., Mladenovic, N., Perez-Britos, D.: Variable neighborhood decomposition search. Journal of Heuristics 7(4), 335–350 (2001)
13. Harvey, W., Ginsberg, M.: Limited Discrepancy Search. In: IJCAI 1995, pp. 607–614 (1995)
14. `http://www.cs.nott.ac.uk/~tec/NRP/`
15. Ikegami, A., Niwa, A.: A subproblem-centric model and approach to the nurse scheduling problem. Mathematical Programming 97(3), 517–541 (2003)
16. Karp, R.: Reducibility among combinatorial problems. In: Complexity of Computer Computations, pp. 85–103. Plenum Press, New York (1972)
17. Li, H., Lim, A., Rodrigues, B.: A hybrid AI approach for nurse rostering problem. In: SAC, pp. 730–735 (2003)
18. Loudni, S., Boizumault, P.: Combining VNS with constraint programming for solving anytime optimization problems. EJOR 191(3), 705–735 (2008)
19. Menana, J., Demassey, S.: Sequencing and counting with the multicost-regular constraint. In: van Hoeve, W.J., Hooker, J.N. (eds.) CPAIOR 2009. LNCS, vol. 5547, pp. 178–192. Springer, Heidelberg (2009)
20. Métivier, J.-P., Boizumault, P., Loudni, S.: Softening `gcc` and `regular` with preferences. In: SAC 2009, pp. 1392–1396 (2009)
21. Millar, H., Kiragu, M.: Cyclic and non-cyclic sheduling of 12h shift nurses by network programming. EJOR 104(1), 582–592 (1996)
22. Mladenovic, N., Hansen, P.: Variable neighborhood search. Computers & OR 24(11), 1097–1100 (1997)
23. Musa, A., Saxena, U.: Scheduling nurses using goal-programming techniques. IIE transactions 16, 216–221 (1984)
24. Ozkarahan, I.: The zero-one goal programming model of a flexible nurse scheduling support system. In: Int. Industrial Engineering Conference, pp. 436–441 (1989)
25. Pesant, G.: A regular language membership constraint for finite sequences of variables. In: Wallace, M. (ed.) CP 2004. LNCS, vol. 3258, pp. 482–495. Springer, Heidelberg (2004)
26. Petit, T., Régin, J.-C., Bessière, C.: Specific filtering algorithms for over-constrained problems. In: Walsh, T. (ed.) CP 2001. LNCS, vol. 2239, pp. 451–463. Springer, Heidelberg (2001)
27. Qu, R., He, F.: A hybrid constraint programming approach for nurse rostering problems. In: 28th SGAI International Conference on AI, pp. 211–224 (2008)
28. Régin, J.-C.: Generalized arc consistency for global cardinality constraint. In: AAAI 1996, pp. 209–215 (1996)
29. Régin, J.-C., Gomes, C.: The cardinality matrix constraint. In: Wallace, M. (ed.) CP 2004. LNCS, vol. 3258, pp. 572–587. Springer, Heidelberg (2004)
30. Régin, J.-C., Petit, T., Bessière, C., Puget, J.-F.: An original constraint based approach for solving over-constrained problems. In: Dechter, R. (ed.) CP 2000. LNCS, vol. 1894, pp. 543–548. Springer, Heidelberg (2000)
31. Simonis, H.: Models for global constraint applications. Constraints 12(1), 63–92 (2007)
32. Valouxis, C., Housos, E.: Hybrid optimization techniques for the workshift and rest assignment of nursing personnel. A.I. in Medicine 20(2), 155–175 (2000)
33. van Hoeve, W., Pesant, G., Rousseau, L.-M.: On global warming: Flow-based soft global constraints. Journal of Heuristics 12(4-5), 347–373 (2006)