Hierarchically nested convex VCSP*

Martin C. Cooper[1] and Stanislav Živný[2]

[1] IRIT, University of Toulouse III, 31062 Toulouse, France
[2] University College, University of Oxford, UK
cooper@irit.fr standa.zivny@cs.ox.ac.uk

Abstract. We introduce tractable classes of VCSP instances based on convex cost functions. Firstly, we show that the class of VCSP instances satisfying the *hierarchically nested convexity property* is tractable. This class generalises our recent results on VCSP instances satisfying the *non-overlapping convexity property* by dropping the assumption that the input functions are non-decreasing [3]. Not only do we generalise the tractable class from [3], but also our algorithm has better running time compared to the algorithm from [3]. We present several examples of applications including soft hierarchical global cardinality constraints, useful in rostering problems. We go on to show that, over Boolean domains, it is possible to determine in polynomial time whether there exists some subset of the constraints such that the VCSP satisfies the hierarchically nested convexity property after renaming the variables in these constraints.

1 Preliminaries

VCSPs As usual, we denote by $\mathbb{N}$ the set of positive integers with zero, and by $\mathbb{Q}$ set of all rational numbers. We denote $\overline{\mathbb{Q}} = \mathbb{Q} \cup \{\infty\}$ with the standard addition operation extended so that for all $\alpha \in \overline{\mathbb{Q}}$, $\alpha + \infty = \infty$.

In a VCSP (Valued Constraint Satisfaction Problem) the objective function to be minimised is the sum of cost functions whose arguments are subsets of arbitrary size of the variables $v_1, \ldots, v_n$ where the domain of v_i is D_i. For notational convenience, we interpret a solution x (i.e. an assignment to the variables $v_1, \ldots, v_n$) as the set of $\langle$variable,value$\rangle$ assignments $\{\langle v_i, x_i\rangle : i = 1, \ldots, n\}$. The range of all cost functions is $\overline{\mathbb{Q}}$.

Network flows Here we review some basics on flows in graphs. We refer the reader to the standard textbook [1] for more details. We present only the notions and results needed for our purposes. In particular, we deal with only integral flows. Let $G = (V, A)$ be a directed graph with vertex set V and arc set A. To each arc $a \in A$ we assign a *demand/capacity* function $[d(a), c(a)]$ and a *weight* (or

* Martin Cooper is supported by ANR Projects ANR-10-BLAN-0210 and ANR-10-BLAN-0214. Stanislav Živný is supported by a Junior Research Fellowship at University College, Oxford.

cost) function $w(a)$, where $d(a), c(a) \in \mathbb{N}$ and $w(a) \in \mathbb{Q}$. Let $s, t \in V$. A function $f : A \to \mathbb{N}$ is called an $s-t$ *flow* (or just a flow) if for all $v \in V \setminus \{s, t\}$,

$$\sum_{a=(u,v)\in A} f(a) = \sum_{a=(v,u)\in A} f(a) \qquad \text{(flow conservation)}.$$

We say that a flow is *feasible* if $d(a) \leq f(a) \leq c(a)$ for each $a \in A$. We define the *value* of flow f as $val(f) = \sum_{a=(s,v)\in A} f(a) - \sum_{a=(v,s)\in A} f(a)$. We define the *cost* of flow f as $\sum_{a\in A} w(a)f(a)$. A *minimum-cost flow* is a feasible flow with minimum cost.

Algorithms for finding the minimum-cost flow of a given value are well known [1]. We consider a generalisation of the minimum-cost flow problem. To each arc $a \in A$ we assign a convex weight function w_a. In particular, we consider the model in which the weight functions w_a ($a \in A$) are convex piecewise linear and given by the breakpoints (which covers the case of convex functions over the integers). We define the *cost* of flow f as $\sum_{a\in A} w_a(f(a))$. The corresponding problem of finding a minimum-cost integral flow is known as the *minimum convex cost flow* problem. In a network with n vertices and m edges with capacities at most U, the minimum convex cost flow problem can be solved in time $O((m \log U)SP(n, m))$, where $SP(n, m)$ is the time to compute a shortest directed path in the network [1].

2 Hierarchically nested convex

A discrete function $g : \{0, \ldots, s\} \to \overline{\mathbb{Q}}$ is called *convex on the interval* $[l, u]$ if g is finite-valued on the interval $[l, u]$ and the derivative of g is non-decreasing on $[l, u]$, i.e. if $g(m + 2) - g(m + 1) \geq g(m + 1) - g(m)$ for all $m = l, \ldots, u - 2$. For brevity, we will often say that g is *convex* if it is convex on some interval $[l, u] \subseteq [0, s]$ and infinite elsewhere (i.e. on $[0, l - 1] \cup [u + 1, s]$).

Two sets A_1, A_2 are said to be *non-overlapping* if they are either disjoint or one is a subset of the other (i.e. $A_1 \cap A_2 = \emptyset$, $A_1 \subseteq A_2$ or $A_2 \subseteq A_1$). Sets $A_1, \ldots, A_r$ are called *hierarchically nested* if for any $1 \leq i, j \leq r$, A_i and A_j are non-overlapping. If A_i is a set of ⟨variable,value⟩ assignments of a VCSP instance $\mathcal{P}$ and x a solution to $\mathcal{P}$, then we use the notation $|x \cap A_i|$ to represent the number of ⟨variable,value⟩ assignments in the solution x which lie in A_i.

Definition 1. *Let $\mathcal{P}$ be a VCSP instance. Let $A_1, \ldots, A_r$ be hierarchically nested sets of ⟨variable,value⟩ assignments of $\mathcal{P}$. Let s_i be the number of distinct variables occurring in the set of ⟨variable,value⟩ assignments A_i. Instance $\mathcal{P}$ satisfies the* hierarchically nested convexity property *if the objective function of $\mathcal{P}$ can be written as $g(x) = g_1(|x \cap A_1|) + \ldots + g_r(|x \cap A_r|)$ where each $g_i : [0, s_i] \to \overline{\mathbb{Q}}$ ($i = 1, \ldots, r$) is convex on an interval $[l_i, u_i] \subseteq [0, s_i]$ and $g_i(z) = \infty$ for $z \in [0, l_i - 1] \cup [u_i + 1, s_i]$.*

Theorem 1. *Any VCSP instance $\mathcal{P}$ satisfying the hierarchically nested convexity property can be solved in polynomial time.*

In our previous paper [3], we proved a special case of Theorem 1 where all functions g_i $(i = 1, \ldots, r)$ are non-decreasing. We give an algorithm to solve VCSPs satisfying the hierarchically nested convexity property in Section 2.1 and a proof of polynomial-time complexity of this algorithm in Section 2.2.

Observe that the addition of any unary cost function cannot destroy the hierarchically nested convexity property. This is because for each ⟨variable,value⟩ assignment $\langle v_j, a\rangle$ we can add the singleton $A_i = \{\langle v_j, a\rangle\}$ which is necessarily either disjoint or a subset of any other set A_k (and furthermore the corresponding function $g_i : \{0, 1\} \to \overline{\mathbb{Q}}$ is trivially convex).

Example 1 (Value-based soft GCC). The Global Cardinality Constraint (GCC), introduced by Régin [8], is a generalisation of the AllDifferent constraint. Given a set of n variables, the GCC specifies for each domain value d a lower bound l_d and an upper bound u_d on the number of variables that are assigned value d. The AllDifferent constraint is the special case of GCC with $l_d = 0$ and $u_d = 1$ for every d. Soft versions of the GCC have been considered by van Hoeve et al. [6].

The value-based soft GCC minimises the number of values below or above the given bound. We show that the value-based soft GCC satisfies the hierarchically nested convexity property.

For every domain value $d \in D$, let $A_d = \{\langle v_i, d\rangle : i = 1, \ldots, n\}$. Clearly, $A_1, \ldots, A_s$ are disjoint, where $s = |D|$. For every d, let

$$g_d(m) \;=\; \begin{cases} l_d - m & \text{if } m < l_d \\ 0 & \text{if } l_d \le m \le u_d \\ m - u_d & \text{if } m > u_d \end{cases}$$

From the definition of g_d, $g_d(m+1) - g_d(m)$ for $m = 0, \ldots, n-1$ is the sequence $-1, \ldots, -1, 0, \ldots, 0, 1, \ldots, 1$. Therefore, for every d, g_d has a non-decreasing derivative and hence is convex.

Example 2 (Nurse Rostering). In a nurse rostering problem, we have to assign several nurses to each shift [2]. There may be strict lower and upper bounds l_i, u_i on the number of nurses assigned to shift i. For example, assigning zero nurses to a shift is no doubt unacceptable. There is also a penalty if we assign too few or too many nurses to the same shift. The cost function is not necessarily symmetric. For example, being short-staffed is potentially dangerous (and hence worse) than being over-staffed which just costs more money. The cost function for shift i could, for example, be $g(z) = \frac{l_i}{z} - 1$ for $0 \le z < l_i$, $g(z) = 0$ for $z \in [l_i, u_i]$ and $g(z) = z - u_i$ for $z > u_i$. It is easily verified that this function is convex.

Example 3 (Hierarchically nested value-based soft GCC). Being able to nest GCC constraints is useful in many staff assignment problems where there is a hierarchy (e.g. senior manager-manager-personnel, foreman-worker, senior nurse-nurse) [9]. We might want to impose soft convex constraints such as each day we

prefer that there are between 10 and 15 people at work, of which at least 5 are managers among whom there is exactly 1 senior manger, with convex penalties if these constraints do not hold.

Suppose that the constraints of a VCSP instance consist of soft GCC constraints on pairwise non-overlapping sets of variables $S_1, \ldots, S_t$. Let $A_{id} = \{\langle x, d\rangle : x \in S_i\}$. Clearly, the sets of assignments A_{id} are hierarchically nested and, as shown in Example 1, the cost functions corresponding to each GCC constraint are convex.

2.1 Algorithm

Our algorithm is similar to the algorithm presented in [3] based on finding a minimum-cost flow in a network. We use a similar network, with the difference that we only require a single arc between any pair of nodes and the corresponding cost function g_i is now an arbitrary convex function (which is not necessarily non-decreasing). Somewhat surprisingly, this small generalisation allows us to solve many more problems, as we have demonstrated in Section 2 since all these examples involve cost functions g_i which are not monotone non-decreasing.

We call the sets A_i $(i = 1, \ldots, r)$ assignment-sets. We assume that the assignment-sets A_i are distinct, since if $A_i = A_j$ then these two sets can be merged by replacing the two functions g_i,g_j by their sum (which is necessarily also convex). Note that the assignment-set consisting of all variable-value assignments, if present in $\mathcal{P}$, can be ignored since it is just a constant. We say that assignment-set A_k is the *father* of assignment-set A_i if it is the minimal assignment-set which properly contains A_i, i.e. $A_i \subset A_k$ and $\nexists A_j$ such that $A_i \subset A_j \subset A_k$. It follows from the definition of hierarchically nested convexity that A_k is unique and hence that the father relation defines a tree. Moreover, again from the definition of hierarchically nested convexity, for every variable v_i of $\mathcal{P}$ and every $a \in D_i$, there is a unique minimal assignment-set containing $\langle v_i, a\rangle$. Indeed, we can assume without loss of generality that this is precisely $\{\langle v_i, a\rangle\}$.

We construct a directed graph $G_{\mathcal{P}}$ whose minimum-cost integral flows of value n are in one-to-one correspondence with the solutions to $\mathcal{P}$. $G_{\mathcal{P}}$ has the following nodes:

1. the source node s;
2. a variable node v_i $(i = 1, \ldots, n)$ for each variable of $\mathcal{P}$;
3. an assignment node $\langle v_i, d\rangle$ $(d \in D_i,\ i = 1, \ldots, n)$ for each possible variable-value assignment in $\mathcal{P}$;
4. an assignment-set node A_i $(i = 1, \ldots, r)$ for each assignment-set in $\mathcal{P}$;
5. the sink node t.

$G_{\mathcal{P}}$ has the following arcs:

1. $a = (s, v_i)$ for each variable v_i of $\mathcal{P}$; $d(a) = c(a) = 1$ (this forces a flow of exactly 1 through each variable node v_i); $w(a) = 0$;

2. $a = (v_i, \langle v_i, d\rangle)$ for all variables v_i and for each $d \in D_i$; $d(a) = 0$; $c(a) = 1$; $w(a) = 0$;
3. $a = (\langle v_i, d\rangle, A_j)$ for all variables v_i and for each $d \in D_i$, where A_j is the minimal assignment-set containing $\langle v_i, d\rangle$; $d(a) = 0$; $c(a) = 1$; $w(a) = 0$;
4. for each assignment-set A_i with father A_j, there is an arc a from A_i to A_j with cost function g_i, demand $d(a) = l_i$ and capacity $c(a) = u_i$.

Clearly, $G_{\mathcal{P}}$ can be constructed from $\mathcal{P}$ in polynomial time. We now prove that minimum-cost flows f of value n in $G_{\mathcal{P}}$ are in one-to-one correspondence with assignments in $\mathcal{P}$ and, furthermore, that the cost of f is equal to the cost in $\mathcal{P}$ of the corresponding assignment.

All feasible flows have value n since all n arcs (s, v_i) leaving the source have both demand and capacity equal to 1. Flows in $G_{\mathcal{P}}$ necessarily correspond to the assignment of a unique value x_i to each variable v_i since the flow of 1 through node v_i must traverse a node $\langle v_i, x_i\rangle$ for some unique $x_i \in D_i$. It remains to show that for every assignment $x = \{\langle v_1, x_1\rangle, \ldots, \langle v_n, x_n\rangle\}$ which is feasible (i.e. whose cost in $\mathcal{P}$ is finite), there is a corresponding minimum-cost feasible flow f in $G_{\mathcal{P}}$ of cost $g(x) = g_1(|x \cap A_1|) + \ldots + g_r(|x \cap A_r|)$.

For each arc a which is incoming to or outgoing from $\langle v_i, d\rangle$ in $G_{\mathcal{P}}$, let $f(a) = 1$ if $d = x_i$ and 0 otherwise. By construction, each assignment-set node A_i in $G_{\mathcal{P}}$ only has outgoing arcs to its father assignment-set. The flow f_a in arc a from A_i to its father assignment-set A_j is uniquely determined by the assignment of values to variables in the solution x. Trivially this is therefore a minimum-cost flow corresponding to the assignment x. The cost of flow f is clearly $\sum_i g_i(|x \cap A_i|)$ which corresponds precisely to the cost of the assignment x.

We remark that since our construction is projection-safe [7], it can be used for Soft Global Arc Consistency for hierarchically nested convex constraints.

2.2 Complexity

Let $\mathcal{P}$ be a VCSP instance with n variables, each with a domain of size at most d, and r assignment-sets A_i. The maximum number of distinct non-overlapping sets A_i is $2nd - 1$ since the sets of assignments A_i form a tree with at most nd leaves (corresponding to single $\langle$variable,value$\rangle$ assignments) and in which all non-leaf nodes have at least two sons. Thus $r = O(nd)$. The network $G_{\mathcal{P}}$ has $n' = O(n + nd + r) = O(nd)$ vertices and arcs. $G_{\mathcal{P}}$ can be built in $O((nd)^2)$ time in a top-down manner, by adding assignment-sets in inverse order of size (which ensures that an assignment-set is always inserted after its father) and using a table $T[\langle v, a\rangle]$=smallest assignment set (in the tree being built) containing $\langle v, a\rangle$.

In a network with n' vertices and m' arcs with capacities at most U, the minimum convex cost flow problem can be solved in time $O((m \log U)SP(n', m'))$, where $SP(n', m')$ is the time to compute a shortest directed path in the network with n' vertices and m' edges [1]. Using Fibonacci heaps [4], $SP(n', m') = O(m' + n' \log n') = O(nd \log(nd))$, since the number of vertices n' and arcs m' are both $O(nd)$. The maximum capacity U in the network $G_{\mathcal{P}}$ is at most n. Hence

an optimal solution to a hierarchically nested convex VCSP can be determined in $O((nd \log n)(nd \log(nd))) = O((nd)^2(\log n)(\log n + \log d))$ time.

The running time of our algorithm is better than the running time of the algorithm from [3], which is $O(n^3d^2)$. The improvement is mostly due to the fact that the new construction involves only $O(nd)$ arcs as opposed to $O((nd)^2)$ arcs in [3]. Moreover, our algorithm solves a bigger class of problems compared to [3]. Overall, we solve more and faster!

3 Renamable Boolean hierarchically nested convex VCSP

In this section we extend the class of hierarchically nested convex VCSPs to allow renaming of certain variables in the case of Boolean domains.

We begin by illustrating the notion of renaming by means of an example. First, we require some notation.

Cost function $\textsc{AtMost}_r(A)$ returns 0 if x contains at most r assignments from the set of assignments A, and $\textsc{AtMost}_r(A)$ returns 1 otherwise. Similarly, cost function $\textsc{AtLeast}_r(A)$ returns 0 if x contains at least r assignments from the set of assignments A, and $\textsc{AtLeast}_r(A)$ returns 1 otherwise. Note that cost functions $\textsc{AtLeast}_1$ and $\textsc{AtMost}_r$, where $r = |A| - 1$, are both convex on $[0, |A|]$. In the remainder of this section we will consider only Boolean VCSPs.

Example 4. Let $\mathcal{P}$ be a Max-SAT instance given in CNF form by the following clauses:

$$(a \vee b \vee c), \quad (c \vee d), \quad (\neg c \vee \neg d \vee e), \quad (\neg a \vee \neg e).$$

Clearly, a clause with literals A can be written as $\textsc{AtLeast}_1(A)$. Notice that, in this example, the first two clauses are overlapping. However, we can replace the second clause by the equivalent constraint $\textsc{AtMost}_1(\{\neg c, \neg d\})$. This gives us an equivalent problem with the following constraints:

$$(a \vee b \vee c), \quad \textsc{AtMost}_1(\{\neg c, \neg d\}), \quad (\neg c \vee \neg d \vee e), \quad (\neg a \vee \neg e).$$

Now $\mathcal{P}$ is expressed as an instance satisfying the hierarchically nested convexity property on the hierarchically nested sets of assignments $\{a, b, c\}$, $\{\neg c, \neg d\}$, $\{\neg c, \neg d, e\}$, $\{\neg a, \neg e\}$.

Example 4 leads to the following definitions:

Definition 2. *Given a valued constraint in the form of the cost function $g(|x \cap A|)$, where A is a set of Boolean assignments (i.e. literals) of size m, we define the* renaming *of this valued constraint, on the set of Boolean assignments denoted by $rename(A) = \bar{A}$, as the valued constraint $g'(|x \cap \bar{A}|) = g(m - |x \cap \bar{A}|) = g(|x \cap A|)$, where $\bar{A} = \{\neg x \mid x \in A\}$.*

The function $g'(z) = g(m - z)$ is clearly convex if and only if g is convex.

Definition 3. *A Boolean VCSP instance $\mathcal{P}$ with the objective function $g_1(|x \cap A_1|) + \ldots + g_r(|x \cap A_r|)$ is* renamable hierarchically nested convex *if there is a subset of the constraints of $\mathcal{P}$ whose renaming results in an equivalent VCSP instance $\mathcal{P}'$ which is hierarchically nested convex.*

Theorem 2. *The class of renamable hierarchically nested convex VCSPs is recognisable and solvable in polynomial time.*

Proof. We show that recognition is polynomial-time by a simple reduction to 2-SAT, a well-known problem solvable in polynomial time [5]. Let $\mathcal{P}$ be a Boolean VCSP instance with r constraints such that the ith constraint ($i = 1, \ldots, r$) is $g_i(|x \cap A_i|)$ for a convex function g_i. For each constraint in $\mathcal{P}$, there is a Boolean variable ren_i indicating whether or not the ith constraint is renamed. For each pair of distinct $i, j \in \{1, \ldots, r\}$, we add clauses of length 2 as follows:

1. if A_i and A_j overlap then add constraint $ren_i \Leftrightarrow \neg ren_j$ (since we must rename just one of the two constraints);
2. if $rename(A_i)$ and A_j overlap then add constraint $ren_i \Leftrightarrow ren_j$ (to avoid introducing an overlap by a renaming).

It is easy to see that solutions to the constructed 2-SAT instance correspond to valid renamings of $\mathcal{P}$ which give rise to an equivalent VCSP instance satisfying the hierarchically nested convexity property. Tractability of solving the resulting instance follows directly from Theorem 1. □

4 Maximality of hierarchically nested convex

This section shows that relaxing either convexity or hierarchical nestedness leads to intractability.

Proposition 1. *The class of VCSP instances whose objective function is of the form $g(x) = g_1(|x \cap A_1|) + \ldots + g_r(|x \cap A_r|)$ where the functions g_i are convex, but the sets of assignments A_i may overlap, is NP-hard, even if $|A_i| \leq 2$ for all $i \in \{1, \ldots, r\}$ and all variables are Boolean.*

Proof. It suffices to demonstrate a polynomial-time reduction from the well-known NP-hard problem Max-2SAT [5]. We have seen in Section 3 that any Max-2SAT clause $l_1 \vee l_2$ (where l_1, l_2 are literals) is equivalent to the $\{0, 1\}$-valued convex cost function $\textsc{AtLeast}_1(|x \cap \{l_1, l_2\}|)$. It is therefore possible to code any instance of Max-2SAT using convex cost functions (on possibly overlapping sets of assignments). □

Proposition 2. *The class of VCSP instances whose objective function is of the form $g(x) = g_1(|x \cap A_1|) + \ldots + g_r(|x \cap A_r|)$ where the sets of assignments A_i are hierarchically nested, but the functions g_i are not necessarily convex, is NP-hard even if $|A_i| \leq 3$ for all $i \in \{1, \ldots, r\}$ and all variables are Boolean.*

Proof. We give a polynomial-time reduction from the well-known NP-complete problem 3SAT [5]. Let I_{3SAT} be an instance of 3SAT with m clauses. The constraint $\textsc{AllEqual}(l_1, l_2, l_3)$ (where l_1, l_2, l_3 are literals) is equivalent to the (non-convex) cost function $g(|x \cap \{l_1, l_2, l_3\}|)$ where $g(0) = g(3) = 0$ and $g(1) = g(2) = \infty$. For each variable v in I_{3SAT}, we use the following gadget G_v based

on non-overlapping AllEqual constraints to produce multiple copies $v_1, \ldots, v_m$ of the variable v and multiple copies $w_1, \ldots, w_m$ of its negation $\overline{v}$: G_v consists of the constraints AllEqual$(\overline{u_i}, \overline{v_i}, \overline{y_i})$ $(i \in \{1, \ldots, m\})$, AllEqual$(y_i, \overline{w_i}, u_{i+1})$ $(i \in \{1, \ldots, m-1\})$, and AllEqual$(y_m, \overline{w_m}, u_1)$, where the variables u_i, y_i only occur in the gadget G_v. It is easy to verify that G_v imposes $v_1 = \ldots = v_m = \overline{w_1} = \ldots = \overline{w_m}$. Furthermore, the variables v_i, w_i only occur negatively in G_v. We now replace the ith clause of I_{3SAT} by a clause in which each positive variable v is replaced by its ith copy v_i and each negative variable $\overline{v}$ is replaced by the ith copy w_i of $\overline{v}$. This produces a hierarchically nested VCSP instance which is equivalent to I_{3SAT} (but whose cost functions are not all convex). □

5 Conclusions

The complexity of the recognition problem for hierarchically nested convex VCSPs is an open problem if the functions g_i are not explicitly given. The complexity of hierarchically nested non-convex VCSPs where all assignment-sets are of size at most 2 is open as well. (Note that the NP-hardness reduction in the proof of Proposition 2 requires assignment-sets of size up to three.)

Acknowledgments We are grateful to Jean-Philippe Métivier for pointing out the utility of soft hierarchically nested GCC in nurse rostering.

References

1. Ahuja, R., Magnanti, T., Orlin, J.: Network Flows: Theory, Algorithms, and Applications. Prentice Hall/Pearson (2005)
2. Burke, E.K., Causmaecker, P.D., Berghe, G.V., Landeghem, H.V.: The state of the art of nurse rostering. Journal of Scheduling 7(6), 441–499 (2004)
3. Cooper, M.C., Živný, S.: Hybrid tractability of valued constraint problems. Artificial Intelligence 175(9-10), 1555–1569 (2011)
4. Fredman, M.L., Tarjan, R.E.: Fibonacci heaps and their uses in improved network optimization algorithms. Journal of the ACM 34(3), 596–615 (1987)
5. Garey, M., Johnson, D.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman (1979)
6. van Hoeve, W.J., Pesant, G., Rousseau, L.M.: On global warming: Flow-based soft global constraints. Journal of Heuristics 12(4-5), 347–373 (2006)
7. Lee, J.H.M., Leung, K.L.: Towards efficient consistency enforcement for global constraints in weighted constraint satisfaction. In: Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI'09). pp. 559–565 (2009)
8. Régin, J.C.: Generalized Arc Consistency for Global Cardinality Constraint. In: Proceedings of the 13th National Conference on AI (AAAI'96). vol. 1, pp. 209–215 (1996)
9. Zanarini, A., Pesant, G.: Generalizations of the global cardinality constraint for hierarchical resources. In: Proceedings of the 4th International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CPAIOR'07). Lecture Notes in Computer Science, vol. 4510, pp. 361–375. Springer (2007)