An Elegant and Efficient Implementation of Russian Dolls Search for Variable Weighted CSP

Thierry Benoist[1,2] and Michel Lemaître[3]

[1] Bouygues e-lab, 1 av. Eugène Freyssinet,
78061 St Quentin en Yvelines Cedex, France
[2] Laboratoire Informatique d'Avignon CNRS FRE 2487,
339 chemin des Meinajaries, Avignon Cedex, 84911, France
tbenoist@bouygues.com
[3] ONERA Center of Toulouse
2 av. Edouard Belin, BP 4025, 31055 Toulouse Cedex 4, France
Michel.Lemaitre@cert.fr

Abstract. The *Russian Dolls Search* (RDS) algorithm is both a solving procedure and a cost based filtering algorithm suitable for *Weighted Constraint Satisfaction Problems* (WCSP). In the context of Variable Weighted CSP, we propose a declarative implementation of RDS filtering, using a set of auxiliary variables and constraints. This approach allows relying on a *Constraint Programming* (CP) solver instead of developing dedicated machinery. We prove that this model yields strictly better propagation than the standard one and we bring experimental results showing that both computation time and number of explored nodes are significantly decreased.

Keywords: Russian Dolls Search, Variable Weighted Constraint Satisfaction Problem, Constraint Programming.

1. Introduction

The *Valued Constraint Satisfaction Problem* (VCSP) framework is an extension of the Constraint Satisfaction Problem (CSP) framework, allowing taking into account costs, preferences and priorities. Hence, it is a convenient framework for constraint optimization [9]. Following [3], we regard *Weighted CSP* (WCSP) as a specific subclass of VCSP for which the aggregation operator is the ordinary sum. In this paper, we consider a further specialization of WCSP, namely the *Variable* WCSP (VWCSP) framework, dedicated to the modeling of knapsack-like problems.

A VWCSP has a subset of 0-1 special variables called the selection variables. Each selection variable X_i is given a weight φ_i, and is associated to a unary soft constraint ($Xi = 1$) of weight φ_i. There is no other soft constraint. All other constraints are hard constraints. Moreover, the following property is required: when a selection variable Xi takes the value 0 (the rejection value), all hard constraints involving X_i must be satisfied.

The standard WCSP objective (minimizing the sum of the unsatisfied constraints) is equivalent, in a VWCSP, to maximizing the sum of the weights of the selection variables which are not instantiated to the rejection value. This framework covers a wide range of applications. Typically most knapsack-like problems consisting in selecting a subset of maximal weight among a given set of items subject to some hard constraints can be seen as VWCSP.

The Russian Dolls algorithm [10] was first developed within this framework in order to optimize the schedule of an Earth Observation Satellite [1]. The idea is to successively solve growing nested sub-problems, starting by the schedule of the very last tasks (photos) and ending by solving the whole problem. Each sub-problem provides a good bound boosting the resolution of the next ones, what makes the whole process much faster than a direct resolution of the whole problem.

This approach consists of a solving procedure considering nested sub-problems one after another, and a dedicated cost based filtering mechanism (based on computed bounds) aside of standard propagation of constraints of the problem. RDS is usually combined with Forward Checking in a standalone WCSP optimizer. Alternatively RDS can be integrated in a CP framework, embedding this filtering algorithm in a global constraint [4]. This last approach presents numerous advantages: low level tasks like events management on variables is supported by the CP kernel, expressive and efficient (global) constraints are available and the objective function can be freely defined. Embedding "soft" filtering in constraints was also a goal pursued by [7].

After a brief description of the original model we will propose, in the context of VWCSP, a declarative implementation of RDS filtering, using a set of auxiliary variables and constraints. This allows relying on the CP solver instead of developing parallel dedicated machinery to perform RDS filtering what can be a tricky task [4]. Incidentally, the whole procedure can be formulated within this model as a more classical tree search. Besides we prove that this model yields better propagation than the standard one and we bring experimental results showing that both computation time and number of explored nodes are significantly decreased.

2. Definitions and notations

2.1. Framework

A *Variable Weighted Constraint Satisfaction Problem* (VWCSP) is defined as follows. We give here a Constraint Programming formulation of this problem.

- *Variables*: problem P involves n+m+1 variables.
  - n binary variables ($X_1 \ldots X_n$) with domain(X_i)$\subseteq\{0,1\}$ (selection variables),
  - m integer variables ($Z_1 \ldots Z_m$)
  - one objective variable Ω
- *Constraints*:
  - Ω is involved in a single constraint $\Omega = \sum \varphi_i X_i$ where $\varphi_1 \ldots . \varphi_n$ is a vector of non-negative integers
  - others constraints obey the following properties
    - at least one X_i is involved,

    - when one of involved X_i equals 0, the constraint is satisfied[1].
- *Objective*: Find a solution satisfying all constraints and maximizing Ω.

Note that a trivial solution of P is $X_i=0\ \forall i$, $\Omega=0$, and $(Z_1 \ldots Z_m)$ taking arbitrary values. As for most "soft" problems the difficulty lies in optimization rather that satisfaction. This framework is particularly suited to model problems where a maximum number of objects have to be selected subject to a set of constraints: all constraints on objects are only "activated" when their selection variable equals 1. Therefore finding a maximum solution to the VWCSP is selecting the best subset of objects such that $(Z_1 \ldots Z_m)$ can be assigned values satisfying triggered constraints.

2.2. Example: a scheduling problem

In this example, three tasks can be performed on a disjunctive resource. Durations are respectively (1,3,3), earliest starts (1,0,0), latest ends (4,5,4) and revenues (2,4,5). The problem is to select a subset of tasks of maximum revenue. This problem can be set as a VWCSP where X_i models the selection of task i and Z_i represents the starting time of task i when selected. The VWCSP reads as follows.

- *Variables[min,max]*: $X_1[0,1]$,$X_2[0,1]$,$X_3[0,1]$,$Z_1[1,3]$,$Z_2[0,2]$ $Z_3[0,1]$, $\Omega[0,11]$
- *Constraints :*
    - $\Omega = 2X_1 + 4X_2 + 5X_3$
    - $(X_1=0)$ OR $(X_2=0)$ OR $(Z_1 + 1 \leq Z_2)$ OR $(Z_2 + 3 \leq Z_1)$
    - $(X_1=0)$ OR $(X_3=0)$ OR $(Z_1 + 1 \leq Z_3)$ OR $(Z_3 + 3 \leq Z_1)$
    - $(X_2=0)$ OR $(X_3=0)$ OR $(Z_2 + 3 \leq Z_3)$ OR $(Z_3 + 3 \leq Z_2)$

An optimal solution of this scheduling problem is $X_1=1$, $X_2=1$, $X_3=0$, $Z_1=1$, $Z_2=2$, $Z_3=0$, $\Omega=6$.

3. The original algorithm

We describe in this section the original RDS algorithm. Each step of this algorithm consists in computing an upper bound of a part of the objective sum, through the exact resolution of a restricted problem.

We consider the VWCSP defined in section 2.1 We note P_j the sub problem where $X_i=0\ \forall\ i < j$, and ω_j the optimal solution of P_j (by convention $\omega_{n+1}=0$). In example 2.2 P_2 is the problem restricted to tasks 2 and 3 and constraints holding on these tasks. Note that $P_1=P$.

Given an integer $k \leq n$ the Russian Dolls filtering algorithm for frontier k (noted $R_{rds}(P,k)$) is based on the following decomposition of the objective function

[1] In other words, constraints follow the pattern $(X_i = 1\ \forall\ i \in I) \Rightarrow$ C, with I a non-empty subset of $[1,n]$.

$$\Omega = \sum_{i=1}^{k-1} \varphi_i X_i + \sum_{i=k}^{n} \varphi_i X_i \qquad (\mathbf{1})$$

With $V^{\min}$ and $V^{\max}$ respectively denoting the lower and upper bounds of the domain of variable V and using ω_k as an upper bound of the second term of (**1**), two filtering rules are added to the arc-consistency algorithm, in order to take this RDS bound ω_k into account:

$$\Omega^{\max} \le \sum_{i=1}^{k-1} \varphi_i X_i^{\max} + \omega_k \qquad (\mathbf{2})$$

$$\forall i_0 \le k-1 \quad \sum_{\substack{i=1 \\ i \ne i_0}}^{k-1} \varphi_i X_i^{\max} + \varphi_{i_0} X_{i_0}^{\min} + \omega_k \ge \Omega^{\min} \qquad (\mathbf{3})$$

Rule (**2**) constrains the upper bound of the objective variable and rule (**3**) detects variables needed to be instantiated to 1 due the objective lower bound. Given ω_i for all $i > j$, these filtering rules can be added to a constraint propagation engine in order to help solving P_j (thus obtaining ω_j). Notice that using the standard RDS algorithm to solve VWCSP requires some care to prevent unary constraints violations from being counted twice (in the forward-checking bound and in the additional RDS previous optimal costs). Usually a frontier k=1+max{i $\in [j+1,n]$ | $X_i^{\min} = X_i^{\max}$} is maintained and only rules $R_{rds}(P,k)$ associated to this frontier are propagated. Variables beyond this frontier have not been instantiated neither directly (enumeration algorithm) nor indirectly (propagation or forward checking).

The Russian Dolls Search consists in successively solving $P_n, P_{n-1}, P_{n-2}, \ldots P_1$ with this technique instead of directly solving the original problem P_1.

```
function Rds(P)
   {for j in n downto 1
        ω_j := solve(P_j,{ω_i | i > j})
    return ω_1}
```

4. New RDS model

In the new RDS model proposed below, a recursive reformulation of the objective sum allows relying on a CP solver to perform a stronger form of RDS filtering.

We enrich the model with n additional variables $\Omega_1, \Omega_2 \ldots \Omega_n$ and n constraints:

- $\Omega_n = \varphi_n X_n$
- $\forall\ i < n \quad \Omega_i = \varphi_i X_i + \Omega_{i+1}$

Given ω_i for all $i > j$, we also post the constraint $\Omega_i \le \omega_i\ \forall\ i > j$. We note $C_{rds}(P,j)$ this set of additional constraints. The corresponding implicit filtering rules achieved by bound-consistency are:

- $\Omega_i^{max} \le \varphi_i X_i^{max} + \Omega_{i+1}^{max}$ and $\Omega_i^{min} \ge \varphi_i X_i^{min} + \Omega_{i+1}^{min}$

- $\varphi_i X_i^{max} \leq \Omega_i^{max} - \Omega_{i+1}^{min}$ and $\varphi_i X_i^{min} \geq \Omega_i^{min} - \Omega_{i+1}^{max}$
- $\Omega_{i+1}^{max} \leq \Omega_i^{max} - \varphi_i X_i^{min}$ and $\Omega_{i+1}^{min} \geq \Omega_i^{min} - \varphi_i X_i^{max}$

Proposition 1: *Domains consistent with $C_{rds}(P,j)$ are always consistent with $R_{rds}(P,k)$ $\forall k \geq j$.*

Proof: Adding equations $\Omega_i^{max} \leq \varphi_i X_i^{max} + \Omega_{i+1}^{max}$ for $i < k$ we obtain

$$\Omega^{\max} = \Omega_1^{\max} \leq \sum \varphi_i X_i^{\max} + \Omega_k^{\max} \tag{4}$$

Since $k>j$, we can use constraint $\Omega_k \leq \omega_k$ and (**4**) is stronger or equal to (**2**).

For any $i_0 \leq k$-1, adding equations $\varphi_i X_i^{min} \geq \Omega_i^{min} - \Omega_{i+1}^{max}$ for $i \leq i_0$ to equations $\Omega_i^{max} \leq \varphi_i X_i^{max} + \Omega_{i+1}^{max}$ for $i \in [i_0+1, k-1]$ yields

$$\forall i_0 \leq k-1 \quad \varphi_{i_0} X_{i_0}^{\min} + \sum_{i=i_0+1}^{k-1} \varphi_i X_i^{\max} + \Omega_k^{\max} \geq \Omega^{\min} - \sum_{i=1}^{i_0-1} \varphi_i X_i^{\max} \tag{5}$$

Since $\Omega_k^{max} \leq \omega_k$, (**5**) is always at least as strong as (**3**).■

Proposition 2: *Given ω_i for all $i > j$, domains consistent with $R_{rds}(P,$ 1+$max\{i \in [j+1,n] \mid X_i^{min} = X_i^{max}\})$ are not necessary consistent with $C_{rds}(P,j)$.*

Proof: With *5* variables, $\varphi_i=1$ $\forall$i, ω_5=1, ω_4=1, ω_3=1, ω_2=2. Some constraints of P cause X_2=0 and X_5=0 when X_1=1. Trying to solve P_1 we set X_1=1, resulting domains are respectively [1,1],[0,0],[0,1],[0,1][0,0].

Therefore $max\{i \in [j+1,n] \mid X_i^{\min} = X_i^{\max}\}$=5 and the upper bound provided by rule (**2**) is just $\sum X_i^{\max}$=3.

On the contrary $\Omega_2^{max} \leq X_2^{\max} + \Omega_3^{max}$ makes $\Omega_2^{max} \leq 1$ and thus $\Omega_1^{max} \leq 2$: the upper bound on Ω is stronger with C_{rds}. ■

Propositions **1** and **2** prove that the filtering performed by this new RDS model C_{rds} is strictly stronger than the classical algorithmic implementation with a single frontier $R_{rds}(k)$.

5. Implementation notes

Implementing Russian Dolls Search on VWCSP with the new model described in section 4 not only provides a strictly stronger filtering but is also much easier to implement. Indeed it relies on filtering rules of linear equations that are available in any constraint solver and it does not require maintaining a frontier between past and future variables. Such an implementation using *n* additional variables and *n* (ternary) linear constraint takes full advantage of the declarativity of constraint programming.

As for the global algorithm it can be implemented as an *n*-ary root choice point in the search tree. Going down the i^{th} branch of this choice point is restricting to problem P_i (setting $X_1 \ldots X_{i-1}$ to 0). Going up this i^{th} branch we register the current best known objective value as an upper bound of Ω_i. Note the classical improvement cut $\Omega \geq lb$+1 where *lb* is the gain of the best solution found so far is automatically handled by the tree exploration mechanism of the CP solver. Together with RDS bound ω_{i+1} and constraint $\Omega_i = \varphi_i X_i + \Omega_{i+1}$ this improvement cut triggers X_i=1 when starting to consider problem P_i (no improving solution can be found without using the new added variable). This expression of a control strategy through the addition of variables and constraints can be compared with the way Local Search exploration was modelled by [6]. This is also similar to the CP-oriented modelling of Max-CSP proposed in [8]

Finally the classical heuristic consisting in first trying to extend the best solution of P_i to a solution of P_{i-1} can be embedded in a "replay" value selection heuristic. This strategy consists in first choosing for variable X_j the value it had in the current best solution. It shall be noted that this heuristic is not specific to Russian Dolls Search: it may be useful in any CP tree search. For instance in a scheduling problem, it may avoid spending time recomputing the daily schedule for Friday if the values of the best current solution are still valid for this day (after changes in the top of the tree, say on Monday). Tending to steer to the best current solution is also the principle of the path-relinking strategy [11].

6. Experimental results

We have compared an implementation of the new model for Russian Dolls Search on VWCSP (called "LightRDS" in the following) with the implementation of the standard (and general) RDS algorithm described in [4]. Both implementations are built on top of the Constraint Programming framework CHOCO [2]. The comparison was done on a set of randomly generated instances of a scheduling problem, namely, the problem of selecting and scheduling the observations of an Earth observing agile satellite. It is a simplified version of a real-world problem, described in [5]. The simplification is still a NP-hard optimisation problem[2]. It is known [10] that the RDS algorithm is particularly well suited for WCSP instances having a small bandwidth constraint graph. This is the case for the instances used in the comparison. Both implemented algorithms use the same static variable and value instantiation orderings. While LightRDS is not constrained to a static variable instantiation algorithm, a dynamic variable instantiation ordering does not improve its performance, at least for the instances used in the comparison. Both algorithms make use of the same initial bounds for the objective value, at the beginning of each RDS step. The figures 1 and 2 show the performances of the algorithms when the size of instances increases. Each point corresponds to the mean CPU time (figure 1) and mean number of created nodes in the search tree (figure 2) for 10 complete resolutions of instances of the same size. Here, the size is the number of candidate observations in the instance.

[2] We thank François Laburthe for a first version of the random instance generator.

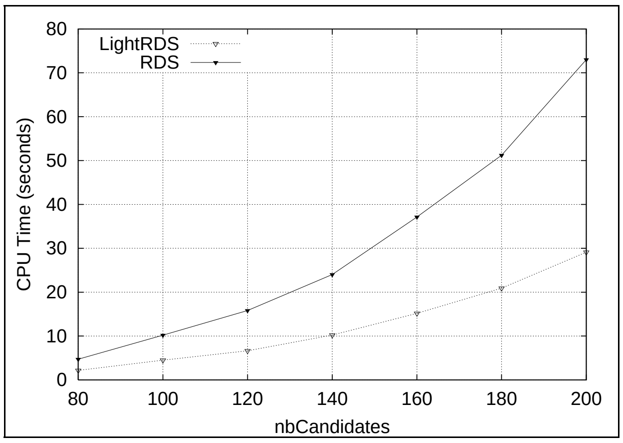

Fig. 1. Comparison of computational times

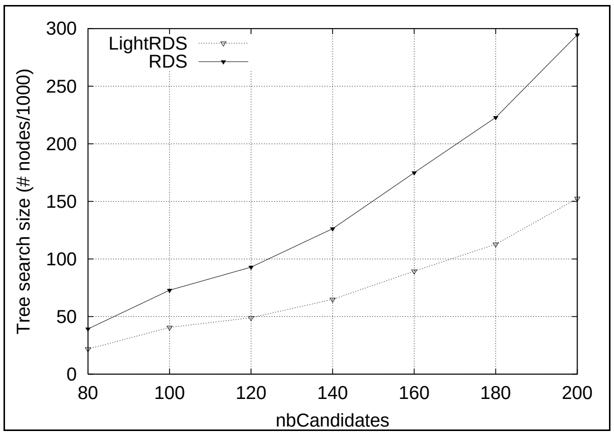

Fig. 2. Comparison of tree search size

The overall result of the comparison is the following: LightRDS is roughly 2.5 times more efficient than the standard RDS, both in terms of CPU times and in number of created nodes. The later result on the size of the search trees confirms that LightRDS does more pruning than standard RDS.

At last, it has been checked that the standard Branch and Bound optimization algorithm provided in CHOCO, using the best ordering heuristics we could find, is not able to compete with the (Light)RDS on this type of problems, which confirms the interest of the RDS scheme on this kind of problems. The figure 3 gives the mean CPU time for 5 complete resolutions using the standard CHOCO Branch and Bound (SCBB) for the same first instances than those used in figures 1 and 2.

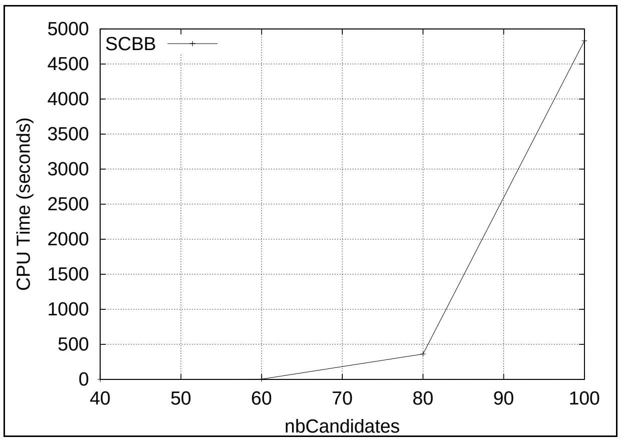

Fig. 3. CPU time without RDS

7. Generalization to WCSP

7.1. Equivalent model

As pointed out in the introduction, the VWCSP framework considered in this paper is a specialization of the WCSP framework: any VWCSP can be reformulated as a WCSP where only unary constraints (equality to the rejection value) are weighted. Conversely, any WCSP can be reformulated into a VWCSP by the following transformation. For each constraint c we add a binary variable X_c and an implication[3] constraint "X_c==1 $\Rightarrow$(c is satisfied)". If φ_c was the penalty[4] of constraint c, then the gain associated to variable X_c is φ_c. Maximizing the total gain of this VWCSP is equivalent to minimizing the sum of penalties in the original WCSP. This equivalence of both frameworks suggests that transposing our new RDS model to WCSP should be an easy task.

Indeed it can be applied on the same static order of variables. If V_1… V_n are the variables of the considered WCSP we can defined P_j as the problem restricted to variables $V_j, V_{j+1} \ldots V_n$, constraints C_i holding on these variables and associated binary variables $\{X_c,\ c \in C_i\}$. With the convention Ω_{n+1}=0 we can define

$$\forall i \le n, \quad \Omega_i = \sum_{c \in C_i - C_{i+1}} \varphi_c X_c + \Omega_{i+1} \qquad \textbf{(6)}$$

Hence the same RDS algorithm can be applied.

[3] Such logical constraints are available in most CP solver.

[4] We assume that no gradual notion of constraint violation is used: constraint c is either satisfied (no penalty) or violated (penalty φ_c).

7.2. Coexistence with Forward-Checking

Although the above generalization of the new model to WCSP is formally correct, we will explain in this section why we believe that it is not an interesting approach in practice. An efficient filtering algorithm dedicated to WCSP is the so-called Forward-Checking (FC) [9]. It is based on the following partition of constraints: $\forall\ i$, F_i is the set of all constraints whose only non-instantiated variable is V_i, F_0 is the set of all constraints with two non-instantiated variables or more (note that it excludes unary constraints from F_0), and F^* is the set of all fully instantiated constraints. It is important to note that this partition is dynamic: at the top of an enumeration tree most constraints belong to F_0, then they enter one of the F_i and end in the F^* set once fully instantiated. The total penalty of the problem is the sum of penalties of each subset and the Forward-Checking filtering is based on the computation of lower bounds of penalties occurring in each subset F_i. For each possible value v of V_i all constraints of F_i are fully instantiated and we note $\varphi_i(v)$ the sum of induced penalties. Thus $lb_i=\min\{\varphi_i(v), v \text{ in } V_i\}$ is a valid lower bound of penalties occurring in F_i. With φ^* the sum of penalties in F^*, the resulting lower bound of the total penalty is $\varphi^*+\sum lb_i$. This lower bound is the base of the Forward-Checking filtering. For instance if $\varphi^*+\sum_{j\neq i} lb_j+\varphi_i(v)$ is larger than the current best cost, then value v can be removed from the domain of V_i.

It shall be noted that both lower bounds (FC and RDS) do not naturally cooperate. With formulation (**6**) both filterings are performed independently. RDS is more efficient at the top of the tree whereas FC works better on nearly instantiated problems, but both algorithms do not interact. In other words RDS bounds are not used inside Forward-Checking. To achieve this cooperation, FC is usually modified as follows: the RDS bound ω_i of P_i is used as a lower bound of penalties of F_0, with P_i the largest RDS sub problem included in F_0. The resulting unified algorithm takes full advantage of both bounds.

It seems difficult to achieve such a tight integration of both views when RDS is statically modeled by constraints (**6**). Indeed due to the dynamicity of partition $\{F_0, F_1,\ldots,F_n,F^*\}$, penalties of F_0 cannot be recursively reformulated with constraints like Ω. In fact no variable can be associated to penalties of F_0 because the evolution of its lower bound is not monotonous along a descent in a search tree (it always ends at 0). Finally we conclude that RDS needs to be algorithmically introduced inside FC to obtain state-of-the-art efficiency on WCSP. Therefore the RDS model proposed in this paper for VWCSP cannot be profitably generalized to WCSP, at least not the way we tried to…

8. Conclusions

In this paper we revisited Russian Dolls Search for Variable Weighted CSP. We described a simple, declarative and static constraint model whose propagation yields a strictly stronger filtering than the original algorithm. This simplification is apparently limited to Variable Weighted CSP but it shall be noted that this VWCSP framework already covers a wide range of applications. It encompasses knapsack problems, prize collecting problems… and all problems whose objective is to select a maximal number of items whatever constraints this selection is submitted to. For such problems constraint programming is often a powerful framework for the design of rich and efficient constraint models. When the small bandwidth of the constraint graph suggests that a Russian Dolls strategy might be efficient, the simplicity of the proposed RDS model allows testing this approach with no need of implementing a dedicated filtering algorithm. Adding a few auxiliary variables and reformulating the objective sum recursively is sufficient to perform an RDS search.

References

1. J. Agnèse, N. Bataille, E. Bensana, D. Blumstein and G. Verfaillie. *Exact and Approximate methods for the Daily Management of an Earth Observation Satellite*. In Proc. of the 5th ESA Workshop on Artificial Intelligence and Knowledge Based Systems for Space 1995.
2. François Laburthe. *Choco: implementing a CP kernel.* In CP00 Post Conference Workshop on Techniques for Implementing Constraint programming Systems (TRICS), Singapore, September 2000.
3. J. Larrosa, T. Schiex. *In the quest of the best form of local consistency for Weighted CSP*. In Proceedings of IJCAI03, Acapulco, Mexico, 2003.
4. M. Lemaître, G. Verfaillie, E. Bourreau, and F. Laburthe. *Integrating algorithms for weighted CSP in a constraint programming framework.* In ICLP'01 - Workshop Proceedings of SOFT'01, Paphos, Cyprus, 26 November-1 December 2001
5. M. Lemaître, G. Verfaillie, F. Jouhaud, J.-M. Lachiver, N. Bataille. *Selecting and scheduling observations of agile satellites.* In Aerospace Science and Technology 6 (2002) 367-381
6. G. Pesant, M. Gendreau. *A view of local search in constraint programming*. In CP-96, pages 353-366, 1996.
7. T. Petit, J.C. Régin and C. Bessière. *Specific Filtering Algorithms of Over-Constrained Problems*. In CP'01, LNCS 2239 pages 451-463, 2001.
8. J.-C. Regin, T. Petit, C. Bessière, J.-F. Puget. *An Original Constraint Based Approach for Solving over Constrained Problems*. In Proceedings of CP00, pages 543-548, Singapore, 2000.
9. T. Schiex, H. Fargier, and G. Verfaillie. *Valued Constraint Satisfaction Problems : Hard and Easy Problems*. In Proc. of IJCAI-95, 631--637, Montréal, Canada, 1995.
10. G. Verfaillie, M. Lemaître, and T. Schiex. *Russian Doll Search for Solving Constraint Optimization Problems*. In Proc. of AAAI-96, 181--187, Portland, OR, 1996.
11. F. Glover, M. Laguna, and R. Martl. *Fundamentals of scatter search and path relinking.* Control and Cybernetics, 39:653-684, 2000.