Multi-Objective Propagation in Constraint Programming

Emma Rollon and **Javier Larrosa** [1]

Abstract. *Bounding constraints* are used to bound the tolerance of solutions under certain undesirable features. Standard solvers propagate them one by one. Often times, it is easy to satisfy them independently, but difficult to satisfy them simultaneously. Therefore, the standard propagation methods fail. In this paper we propose a novel approach inspired in multi-objective optimization. We compute a *multi-objective lower bound set* that, if large enough, can be used to detect the inconsistency of the problem. Our experiments on two domains inspired in real-world problems show that propagation of additive bounding constraints using our approach is clearly superior than previous approaches.

1 Introduction

A *constraint satisfaction problem* (CSP) requires the assignment of a set of variables in such a way that a set of constraints is satisfied. Many problems have constraints that refer to different objectives. Consider the scheduling of a major football league. A good schedule should be *fair* (no team should feel that is treated *much* worse than any other,...), *exciting* (e.g., hot games should be scheduled in accordance to TV requests, several hot games should not take place the same day, ...), *safe* (e.g., rival teams should not play simultaneously in close stadiums, teams with violent followers should not play late in the evening or near a play-ground in the afternoon), *etc*. One way to deal with different criteria is to establish *tolerance bounds* for each one, and find satisfying assignments with respect to all of them. Often, these constraints are in conflict, meaning that assignments that are good with respect one constraint are likely to be bad with respect another. Typically, it is easy to satisfy each constraint independently, but it is hard to satisfy all of them simultaneously. Hence, the difficulty lays on the conjunction.

CSPs are usually solved by searching on the tree of possible assignments. After each assignment a *propagation* process takes place. Its goal is to detect if the current assignment can be extended to a solution. Most propagation algorithms detect and discard domain values that are inconsistent with the current assignment. If some variable loses all its values, the algorithm backtracks. Typically, each constraint is propagated independently. Namely, a value is removed if it is shown to be inconsistent with respect one of the constraints. The only *communication* between constraints is through *value pruning* (pruning one value due to one constraint, may produce the pruning of another value due to another constraint, yielding a cascade effect). This solving approach may not be strong enough for problems with conflicting bounding constraints. Consider the following two constrains over 0-1 variables:

$$x_1x_2 + x_2x_3 + x_3x_4 \geq 1; \quad \sum_{i=1}^{4} x_i \leq 1$$

Note that every value is consistent with respect each constraint, because every value can be extended to satisfy it. However, it is clear that no value is consistent with respect the two simultaneously. This is a simple example that many solvers would deal efficiently with. However, we will show in Section 4 that, if constraints are more intricate, standard solvers may perform poorly.

We propose a more appropriate propagation method. Essentially, our approach consists on considering CSP subproblems as multi-objective minimization problems. The set of tolerance bounds plays the role of a multi-objective upper bound. Then, we compute a multi-objective lower bound that, if large enough, allows backtracking. More precisely, we use *multi-objective mini-bucket elimination* (MO-MBE) [10] as a general parameterizable multi-objective lower bound algorithm. Our experiments on two domains inspired on real world problems show the suitability of our approach.

2 Preliminaries

2.1 CSPs, Bounding Constraints, and MBE

A *constraint satisfaction problem* (CSP) is a triplet $\mathcal{P} = (\mathcal{X}, \mathcal{D}, \mathcal{R})$ where $\mathcal{X}$ is an ordered set of *variables*. Each variable $x_i \in \mathcal{X}$ takes values on a finite *domain* $D_i \in \mathcal{D}$. $\mathcal{R}$ is a set of *constraints*. For convenience, we will consider that a constraint $c \in \mathcal{R}$ is a boolean function over a subset of the variables $var(c) \subseteq \mathcal{X}$ called its *scope*. A tuple t is a (possibly partial) assignment over a subset of the variables $var(t) \subseteq \mathcal{X}$. A singleton tuple is noted $(x_i \leftarrow a)$ and the join of two tuples is noted $t \cdot t'$. If $var(t) = var(c)$, then $c(t)$ tells whether the assignment is allowed by the constraint. We allow partial assignments of functions. Thus, if $var(t) \subset var(c)$, then $c(t)$ is a new constraint in which variables of $var(t)$ have been *fixed* as indicated by t, and its new scope is $var(c) - var(t)$. If $var(t) \not\subseteq var(c)$, then $c(t)$ is the partial assignment of c with respect to $var(t) \cap var(c)$. A *solution* of a CSP is a complete assignment that satisfies all the constraints. Solving a CSP is in general NP-complete.

An *additive bounding constraint* is a pair $(\mathcal{F}, \top)$, where $f \in \mathcal{F}$ are cost functions and $\top \in \mathbb{N}$ is the maximum acceptable cost. For convenience, we define the sum of costs as $u \oplus v = \min\{u + v, \top\}$ Tuple t satisfies the constraint iff,

$$\sum_{f \in \mathcal{F}} f(t) < \top$$

In this paper we will consider CSPs with $p > 1$ additive bounding constraints. Note that we do not make any assumption over cost functions $f \in \mathcal{F}$, which makes the concept of additive bounding

[1] Universitat Politècnica de Catalunya, Spain, email: {erollon, larrosa}@lsi.upc.edu

function $\texttt{MBE}(z, \mathcal{F}, \top)$
  for each $i = n..1$ **do**
    $\mathcal{B}_i := \{f \in F |\, x_i \in var(f)\}$
    $\{\mathcal{P}_{i_1}, \ldots, \mathcal{P}_{i_r}\} :=$ Partition$(z, \mathcal{B}_i)$;
    for each $k = 1..r$ **do** $g_{i_k} := (\sum_{f \in \mathcal{P}_{i_k}} f) \downarrow x_i$;
    $F := (F \cup \{g_{i_1}, \ldots, g_{i_r}\}) - \mathcal{B}_i$;
  endfor
  return g_1;
endfunction

Figure 1. Mini-Bucket Elimination algorithm.

constraint extremely general. Besides, the ideas that we will introduce directly apply to more general bounding constraints such as *semiring-based* [2].

In recent years, many propagation algorithms for additive bounding constraints have been proposed. For instance, all *weighted* CSP local consistencies [9, 3] can be used for this purpose. Another alternative is *mini-bucket elimination* MBE [6], which is the basis of our work. The main advantage of MBE over local consistencies is its control parameter z that allows to trade resources for accuracy. For our purposes in this paper, MBE has the additional advantage of being extendible to multi-objective optimization. The main disadvantages are that MBE cannot be easily used for filtering [5] and that its implementation inside a solver is not incremental. MBE uses the following two operations over cost functions.

- Let f and g be two cost functions. Their sum, denoted $f + g$, is a new function with scope $var(f) \cup var(g)$ defined as,

$$(f + g)(t) = f(t) \oplus g(t)$$

- Let f be a cost function. The elimination of x_i, $f \downarrow x_i$, is a new function with scope $var(f) - \{x_i\}$ defined as,

$$f \downarrow x_i(t) = \min_{a \in D_i} \{f(t \cdot (x_i \leftarrow a))\}$$

Figure 1 shows MBE. It receives an additive bounding constraint $(\mathcal{F}, \top)$ and returns a lower bound of the best assignment. MBE can be used as a propagation procedure inside search, because if the lower bound equals $\top$, the constraint cannot be satisfied and the current line of search can be abandoned. MBE processes variables in decreasing order. For each variable x_i, it computes $\mathcal{B}_i$ the set of cost functions having x_i in the domain. $\mathcal{B}_i$ is partitioned into subsets such that the join scope has size at most $z+1$. The functions of each subset are summed and variable x_i is eliminated. After the last elimination, a zero-arity (namely, a constant) contains the result. MBE is time and space exponential on parameter z.

2.2 Multi-Objective Minimization

Let us consider problems with p objectives. Let $\vec{v}$ and $\vec{u}$ be *two distinct* p-vectors. We say that $\vec{v}$ *dominates* $\vec{u}$ (noted $\vec{v} < \vec{u}$) if $\forall j,\ v_j \le u_j$. A *multiobjective weighted constraint satisfaction problem* (MO-WCSP) is defined as $\mathcal{P} = (\mathcal{X}, \mathcal{D}, \{\mathcal{F}_j, \top_j\}_{j=1}^p)$, where $\mathcal{X}$ and $\mathcal{D}$ are variables and domains. $\mathcal{F}_j$ defines the j-th objective, $F_j(X) = \sum_{f \in \mathcal{F}_j} f(X)$. The multi-objective function is $F(X) = (F_1(X), \ldots, F_p(X)))$. Given a complete assignment X, we say that $F(X)$ is *consistent* iff $\forall j, F_j(X) < \top_j$. A *solution* is a consistent complete assignment. A solution X is *efficient* or *Pareto optimal* if there is no better solution (i.e., $\forall X',\ F(X') \not< F(X)$). $\mathcal{X}_E$ is the set of efficient solutions and $\mathcal{EF}$ is the corresponding *efficient frontier*. The task in a MO-WCSP is to compute $\mathcal{EF}$ (and, possibly, one or all efficient solutions for each of its elements).

Multi-objective MBE (MO-MBE) [10] extends MBE to the multi-objective context. Let $\vec{\top} = (\top_1, \ldots, \top_p)$. The sum of p-vectors is defined as,

$$\vec{v} + \vec{u} = \begin{cases} \vec{\top} & \exists j,\ v_j \oplus u_j = \top_j \\ (v_1 \oplus u_1, \ldots, v_p \oplus u_p) & otherwise. \end{cases}$$

The non-domination closure of a set of p-vectors S is defined as $\langle\ S\ \rangle = \{v \in S\ |\forall u \in S,\ u \not< v\}$. MO-MBE works with p-functions which, instead of scalar costs, return sets of non-dominated p-vectors. For that purpose, original cost functions $f \in \mathcal{F}_j$ need to be reformulated from $f(t) = u$ to $f(t) = \{(0, \ldots, 0, u, 0, \ldots, 0)\}$ (where the u component is at the j-th position). Operations on functions are accordingly extended,

- Let f and g be p-functions. Their sum $h = f + g$ is defined as,

$$h(t) = \langle\ \{\vec{v}|\ t = t' \cdot t'', \vec{v} = \vec{v}' + \vec{v}'', \vec{v}' \in f(t'), \vec{v}'' \in g(t'')\}\ \rangle$$

- Eliminating of x_i from a p-function f, $h = f \downarrow x_i$, is defined as,

$$h(t) = \langle\ \{\vec{v}\ |\ \forall a \in D_i,\ \vec{v} \in f(t \cdot (x_i \leftarrow a))\}\ \rangle$$

Note that, if $p = 1$, these definitions reduce to the classical ones.

Consider as an example the 2-functions f and g in Figure 2 with $\vec{\top} = (15, 18)$ under domains $\{a, b\}$. The sum $f + g$ is a 2-function $(f + g)(x_1, x_2, x_3)$. Note that in $(f + g)(a, b, a)$, the sum of the 2-vectors $(4, 10)$ and $(11, 1)$ is $\vec{\top}$. As $\vec{\top}$ is dominated by $(10, 12)$, it has been removed. The elimination of variable x_3 from $f + g$ is a 2-function $(f + g) \downarrow x_3(x_1, x_2)$. Note that in $(f + g) \downarrow x_3(a, a)$, the 2-vector $(4, 9)$ has been removed as a consequence of the non-domination closure. Moreover, $\vec{\top}$ has also been removed from $(f + g) \downarrow x_3(a, b)$ for the same reason.

The code of MO-MBE is the same as MBE in Figure 1. The only difference is that costs are now p-vectors, and cost functions are now p-functions. Given a MO-WCSP, MO-MBE returns a set S of non-dominated p-vectors which forms a *lower bound set* of its efficient frontier $\mathcal{EF}$ (namely, $\forall \vec{v} \in \mathcal{EF},\ \exists \vec{v}' \in S$ such that either $v = v'$ or $v' < v$). If MO-MBE returns $\vec{\top}$, the MO-WCSP does not have any solution. The time complexity of MO-MBE is $O(e \times \prod_{j=1}^{p-1} \top_j^2 \times d^z)$, where e is the number of p-functions and d is the largest domain size.

3 Propagating Additive Constraints using Multi-Objective Techniques

Let $\mathcal{P} = (\mathcal{X}, \mathcal{D}, \mathcal{R})$ be a CSP such that its set of constraints can be divided into two sets: $\mathcal{H}$ is a set of arbitrary hard constraints, and $\{(\mathcal{F}_j, \top_j)\}_{j=1}^p$ are p additive bounding constraints. The problem is solved with a usual systematic search procedure. Consider an arbitrary search state with its associated partial assignment, which can be seen as if some of the domains have become singletons. Let $\mathcal{P}' = (\mathcal{X}, \mathcal{D}', \mathcal{R})$ be the associated subproblem. At this point, standard solvers would propagate the current assignment using general or specific algorithms for each type of constraint. Since we consider MBE as the propagation algorithm of bounding constraints and it does not prune unfeasible values, it makes sense to propagate in two

f:

x_1	x_2	
a	a	$\{(3,2),(2,8)\}$
a	b	$\{(4,10)\}$
b	a	$\{\vec{\top}\}$
b	b	$\{\vec{\top}\}$

g:

x_2	x_3	
a	a	$\{(1,2)\}$
a	b	$\{(2,1)\}$
b	a	$\{(6,2),(11,1)\}$
b	b	$\{\vec{\top}\}$

$f+g$:

x_1	x_2	x_3	
a	a	a	$\{(4,4),(3,10)\}$
a	a	b	$\{(5,3),(4,9)\}$
a	b	a	$\{(10,12)\}$
a	b	b	$\{\vec{\top}\}$
b	a	a	$\{\vec{\top}\}$
b	a	b	$\{\vec{\top}\}$
b	b	a	$\{\vec{\top}\}$
b	b	b	$\{\vec{\top}\}$

$(f+g)\downarrow x_3$:

x_1	x_2	
a	a	$\{(4,4),(3,10),(5,3)\}$
a	b	$\{(10,12)\}$
b	a	$\{\vec{\top}\}$
b	b	$\{\vec{\top}\}$

Figure 2. Sum and projection over 2-functions. $\vec{\top} = (15, 18)$.

steps: first the set $\mathcal{H}$, and second a sequence of MBE executions, one for each bounding constraint. The practical effectiveness of MBE can be greatly improved if each $\mathcal{F}_j$ is augmented with the set of hard constraints $\mathcal{H}$,

$$F_j(X) = \sum_{f \in \mathcal{F}_j \cup \mathcal{H}} f(X) < \top_j$$

For this purpose, boolean functions $c \in \mathcal{H}$ must be redefined in terms of costs (namely, $c(t) \in \{\top_j, 0\}$). Formally, we can see the propagation process as,

$$\texttt{Propagate}(\mathcal{H}) \wedge \bigwedge_{j=1}^{p} (\texttt{MBE}(z, \mathcal{F}_j \cup \mathcal{H}, \top_j) < \top_j)$$

If the previous expression returns *false* the search procedure should backtrack.

We propose the use of multi-objective mini-buckets instead of mono-objective mini-buckets. It requires the replacement of the sequence of calls to MBE by a single call to MO-MBE,

$$\texttt{Propagate}(\mathcal{H}) \wedge (\texttt{MO-MBE}(z, \{(\mathcal{F}_j \cup \mathcal{H}, \top_j)\}_{j=1}^{p}) < \vec{\top})$$

The previous change may seem a minor modification. However, the subjacent algorithm is completely different and the kind of inference performed is much more powerful, as can be seen in the following example.

Consider a CSP $\mathcal{P} = (\mathcal{X}, \mathcal{D}, \mathcal{R})$ with three 0/1 variables and two bounding constraints: $(F_1, 12)$ with $F_1 = \{f_1(x_1) = 10x_1, f_2(x_2) = 10x_2, f_3(x_3) = 2x_3\}$, and $(F_2, 10)$ with $F_2 = \{h_1(x_1) = 3(1-x_1), h_2(x_2) = 4(1-x_2), h_3(x_3) = 8(1-x_3)\}$. There are two additional constraints $x_1 \neq x_2$ and $x_2 \vee x_3$. If we propagate with MBE($z=2$) each bounding constraint (augmented with hard constraints) we obtain lower bounds 10 and 3, respectively. Find below the trace of each execution (note that in this example, there is no need to break buckets into mini-buckets),

1st bounding constr.:

domain value	g_3	g_2	$g_1()$
1	0	2	10
0	2	10	

2nd bounding constr.:

domain value	g_3	g_2	$g_1()$
1	0	4	3
0	0	0	

The propagation indicates that the problem *may have* solution. Note that each lower bound is the optimum assignment of one bounding constraint discarding the other, so MBE is doing a perfect estimation with the information that it receives. Its problem is that it only *knows* part of the information. If we propagate with MO-MBE($z=2$) the two bounding constraints simultaneously (augmented with hard constraints) we obtain a lower bound set of $\{\vec{\top}\}$ which indicates that the problem *does not have* any solution. Find below the trace of the execution,

domain value	g_3	g_2	$g_1()$
1	$\{(2,0),(0,8)\}$	$\{(2,4)\}$	$\{\vec{\top}\}$
0	$\{(2,0)\}$	$\{(10,8)\}$	

The key is that in each execution of MBE, it searches for one different valid assignment. Each one satisfies one bounding constraint separately, but not simultaneously. However, MO-MBE searches for a unique assignment that satisfies both bounding constraints simultaneously and cannot find it.

4 Computational Experiments

We tested our propagation mechanism in two different domains: *combinatorial auctions* and *satellite scheduling*. In both cases the problem contains two additive bounding constraints with unary cost functions plus a set of small arity hard constraints. We compare the performance of four algorithms based on depth-first search. The first is IlogSolver 6.1 with default parameters. Bounding constraints are encoded with the IloPack global constraint. The second algorithm enforces FDAC [9] with each bounding constraint augmented with the hard constraints. The third one enforces arc-consistency in the set of hard constraints and then executes MBE ($z=2$) with the bounding constraints (namely, the first approach described in Section 3). The fourth one is like the third, but the two calls to MBE are replaced to one call to MO-MBE (namely, the second approach described in Section 3). For comparison, we always report cpu time. We run all the experiments on a Pentium IV at 3GHz with 2 GB of memory, running Linux.

4.1 Combinatorial Auctions

Combinatorial auctions (CA) allow bidders to bid for indivisible subsets of goods. In risk-conscious auctions, the bid-taker wants to control the risk of bid withdrawal following winner determination, because it may cause large losses in revenue [7]. Consider a set of goods $\{1, 2, \ldots, n\}$ that go on auction. There are m bids. Bid i is defined by the subset of requested goods $X_i \subseteq \{1, 2, \ldots, n\}$, the money offer b_i and the probability of payment failure r_i. The bid-taker must decide which bids are to be accepted. We consider a decision version of the problem in which the bid-taker provides two constants (P, R) indicating that she does not want to miss more than P over the maximum possible revenue ($\sum_{i \leq m} b_i$) and probability of bid withdrawal lower than R. Both constraints can be expressed as additive (the second one requires a probabilistic independence assumption and a logarithmic transformation).

We have generated mono-objective CA using the PATH model of the CATS generator [8] and randomly added payment failure probabilities to the bids in the range 0.0 to 0.3. We experiment on instances with 20 and 50 goods, varying the number of bids from 80 to 200. For each parameter configuration, we generate samples of size 25 and set the time limit to 300 seconds. For each instance, we established the values of (P, R) in such a way that $i)$ the instance admits a solution and $ii)$ a small decrease of either one renders the problem unsoluble. Consequently, the instances are difficult with respect the two constraints.

Figure 3 reports the results for instances with 20 and 50 goods, respectively. It can be observed that problems become harder as the number of bids increases. Regarding the algorithms, it is clear that MO-MBE propagation always outperforms the other three approaches. For instances with 20 goods, it is about 6 times faster than its competitors. With 50 goods the gain is still larger (up to 10 times faster).

4.2 Earth observation satellite scheduling.

An earth observation satellite, such as the Spot5 [1], orbits the earth while taking photographs requested by different customers. It is impossible to fulfill all the requests. Thus, the problem is to select the subset that the satellite will actually take and decide which camera will be used for each one. We experiment with Spot5 instances that have binary and ternary hard constraints and variables with domains of size 2 and 4. We consider instances with 2 bounding constraints. The first one comes from the on-board storage limit. The second one comes from the importance of photographs (each photograph has an associated penalty for not taking it). We consider the decision problem in which two constants (S, P) are given: the available on-board memory has size S and cannot be surpassed, and the maximum acceptable aggregated penalty is P. Since we could not solve complete instances, we considered subinstances as follows: $X_{\geq k}$ denotes instance X where photographs whose penalty is less than k have been eliminated.

Figure 4 reports the results for instance $1506_{\geq 1000}$. Since we observed that the behavior of other subinstances (i.e., $1401_{\geq 1000}$, $1403_{\geq 1000}$, $1405_{\geq 1000}$, and $1407_{\geq 1000}$) was very similar, we do not report their results. Each plot reports results for a fixed value of P and varying the value of S. We established a time limit of 600 seconds. Note the logarithmic scale. We observed that IlogSolver always performs very poorly and only solves instances with $S \leq 4$. Thus, we omit it from the plot.

Considering MBE and MO-MBE, we observe the following pattern that is best exemplified in the $P = 450000$ plot (Figure 4 top left). For high values of S, MBE is more efficient than MO-MBE. The reason is that the memory constraint is very easy to satisfy, which makes it practically irrelevant. MBE already captures the difficulty of the problem, which is mono-objective in nature. Thus, the higher overhead of MO-MBE is wasted. As the value of S decreases, the situation changes. Both bounding constraints become difficult to satisfy simultaneously. Propagating with mono-objective MBE fails in detecting inconsistency because it is easy to satisfy each constraint if the other one is disregarded, but it is difficult to satisfy the two of them simultaneously. Only the bi-objective nature of MO-MBE can capture such difficulty. As a result, MBE cannot solve the problems, while MO-MBE solves them in a few seconds. If S decreases even further, the memory constraint becomes clearly unsatisfiable in conjunction with the penalty constraint. MO-MBE propagation detects it easily but MBE propagation does not. Only for the lowest values of S, when the constraint is unsatisfiable independently of other constraints, MBE detects it efficiently. The algorithm that enforces FDAC behaves similarly to MBE because it also considers the two bounding constraints separately. However, it provides a much better average performance.

Observing the plots in decreasing order of P, we observe that problems become harder as the penalty bounding constraint becomes tighter and harder to satisfy. As before, there is a range of S for which the instances are most difficult. This difficulty peak shifts towards the right as P decreases. For MO-MBE propagation, the range is narrower than for MBE and FDAC, but it also fails to solve some instances within the time limit of 600 seconds.

The $P = 250000$ case requires further discussion: the plot only shows the left-hand side of the difficulty peak, where the tight memory constraint *helps* MO-MBE to prove unsatisfiability almost instantly whilst MBE and FDAC cannot. For large values of S the constraint becomes trivial and irrelevant. Then the problem difficulty is given only by the penalty constraint and the three algorithms fail in solving it.

5 Related Work

The idea of using the conjunction of two or more constraints during propagation, rather than using them one-by-one, is not new. For instance, path-consistency, path-inverse consistency and neighborhood inverse consistency [4] use this idea at different levels of sophistication. However, all these works assume binary problems and cannot be efficiently extended to higher arity constraints such as bounding constraints. The work of [12] is also related to ours. However, it is restricted to problems with so-called *knapsack constraints*, which are a special case of pairs of additive bounding constraints that share unary cost functions (namely, linear constraints of the form $L \leq AX \leq U$). A little bit more general is the work of [11], which applies to pairs of constraints of the form,

$$\sum_{i=1}^{n} w_i x_i \leq U \quad \wedge \quad \sum_{i=1}^{n} p_i x_i > U$$

Our notion of additive bounding constraint includes these and many other cases and allow us to take into account any number of bounding constraints. Besides, it can be easily extended to more sophisticated bounding constraints expressable in terms of semirings [2]. Overmore, our algorithmic approach using multi-objective optimization techniques is radically different.

6 Conclusions and Future Work

Additive bounding constraints, $\sum_{f \in \mathcal{F}} f(X) < \top$, are used to bound the tolerance under certain undesirable feature in problem solutions. The propagation in problems involving conflicting bounding constraints is a difficult task for standard solvers. Typically, they propagate constraints one by one. When it is easy to satisfy bounding constraints independently, but difficult to satisfy them simultaneously, this approach clearly fails. In this paper we have proposed a novel approach inspired in multi-objective optimization. We propagate the additive bounding constraints simultaneously with multi-objective mini-bucket elimination MO-MBE [10]. The output is a multi-objective lower bound set that can be used to detect the inconsistency of the problem. Our experiments on two domains inspired in real-world problems show that propagation of additive bounding constraints using MO-MBE is clearly superior than previous approaches.

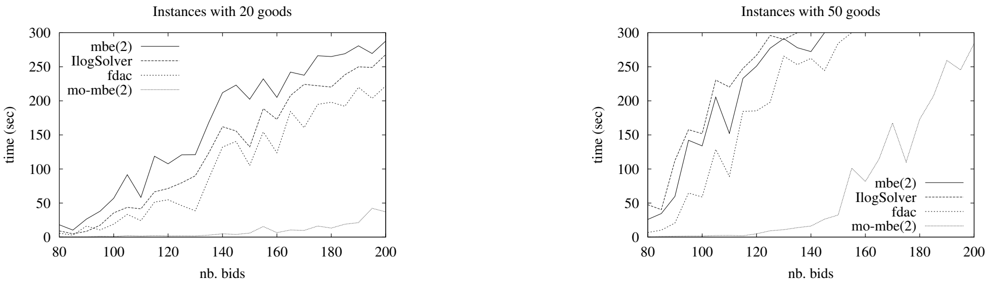

Figure 3. Experimental results on CA for 20 and 50 goods, respectively. Risk probabilities ranging from 0.0 to 0.3. Average time on 25 instances for each parameter configuration. Time limit 300 sec.

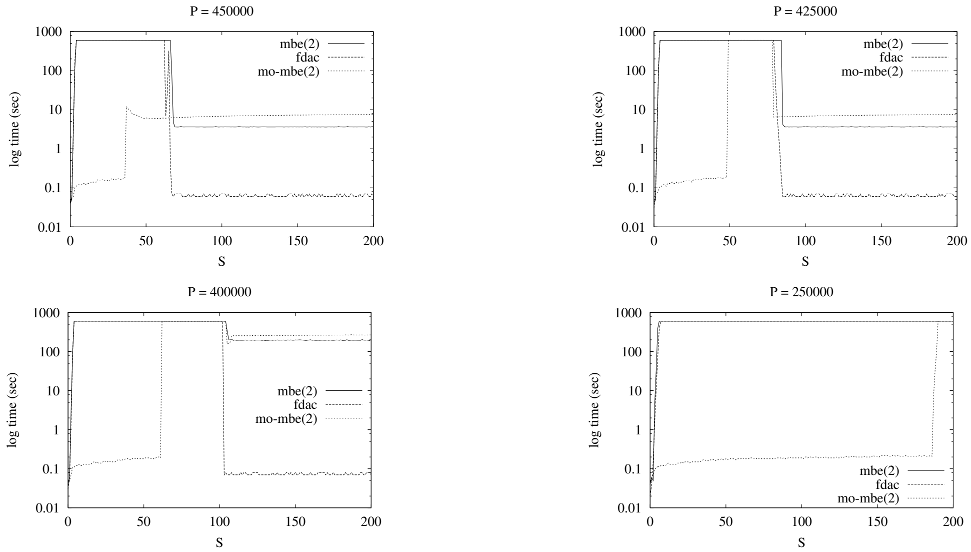

Figure 4. Experimental results on $1506_{\geq 1000}$ spot5 instance. Time limit 600 sec. Note the logarithmic scale.

The high overhead of multi-objective propagation may render it useless in problems with many bounding constraints. In that case, it may be useful to detect automatically pairs of conflicting constraints and apply MO-MBE to these pairs independently. Moreover, the experiments indicated that lose bounding constraints cause overhead but are of no use to our approach, so they should be detected and discarded in the propagation process. The development of this idea is part of our future work. A major drawback of MO-MBE propagation is that it cannot detect and prune unfeasible values. We want to overcome this problem using the ideas of [5].

ACKNOWLEDGEMENTS

This research has been funded with project TIN2005-09312-C03-02.

REFERENCES

[1] E. Bensana, M. Lemaitre, and G. Verfaillie, 'Earth observation satellite management', *Constraints*, **4(3)**, 293–299, (1999).

[2] S. Bistarelli, H. Fargier, U. Montanari, F. Rossi, T. Schiex, and G. Verfaillie, 'Semiring-based CSPs and valued CSPs: Frameworks, properties and comparison', *Constraints*, **4**, 199–240, (1999).

[3] S. de Givry, F. Heras, J. Larrosa, and M. Zytnicki, 'Existential arc consistency: getting closer to full arc consistency in weighted csps', in *Proc. of the 19th IJCAI*, Edinburgh, U.K., (August 2005).

[4] R. Debruyne and C. Bessière, 'Domain filtering consistencies', *Journal of Artificial Intelligence Research*, **14**, 205–230, (2001).

[5] R. Dechter, K. Kask, and J. Larrosa, 'A general scheme for multiple lower bound computation in constraint optimization', in *CP-2001*, pp. 346–360, (2001).

[6] Rina Dechter and Irina Rish, 'Mini-buckets: A general scheme for bounded inference', *Journal of the ACM*, **50**(2), 107–153, (March 2003).

[7] A. Holland, *Risk Management for Combinatorial Auctions*, Ph.D. dissertation, Dept. of Computer Science, UCC, Ireland., 2005.

[8] M.Pearson K.Leuton-Brown and Y.Shoham, 'Towards a universal test suite for combinatorial auction algorithms', *ACM E-Commerce*, 66–76, (2000).

[9] J. Larrosa and T. Schiex, 'In the quest of the best form of local consistency for weighted csp', in *Proc. of the 18th IJCAI*, Acapulco, Mexico, (August 2003).

[10] E. Rollon and J. Larrosa, 'Depth-first mini-bucket elimination', in *Proc. of the 11th CP*, pp. 563–577, Sitges (Spain), (2005). LNCS 3709. Springer-Verlag.

[11] Meinolf Sellmann, 'Approximated consistency for knapsack constraints', in *Proc. of the 9th CP*, pp. 679–693. LNCS 2833. Springer Verlag, (2003).

[12] Michael Trick, 'A dynamic programming approach for consistency and propagation for knapsack constraints', *Annals of Operations Research*, **118**(118), 73–84, (2003).