HOWTO: Asynchronous PFC-MRDAC —Optimization in Distributed Constraint Problems±Adopt—

Marius-Călin Silaghi
Florida Institute of Technology
msilaghi@cs.fit.edu

Abstract

Each agent has private problems. Private concerns can often be formulated in a general framework such as constraint satisfaction (where everything is modeled by either variables, values, or constraints). Often agents need to find agreement with others for the allocation of final resources. The constraint satisfaction (CSPs) is only a special case of optimization. Here it is shown how a very general technique for Distributed CSPs, Replica-based Multiply Asynchronous Search (R-MAS) (comprising ABT,ABTR,AAS,DMAC,DMAC-ABT), can be extended and applied to optimization problems in distributed Weighted CSPs (WCSPs). Centralized WCSPs can be seen as MAX-CSPs where several constraints can link the same variables. PFC-MRDAC is a good approach to MAX-CSPs. How to asynchronize and adapt it to distributed WCSPs? This article describes how asynchronous consistency maintenance can be introduced in Adopt and in Asynchronous Branch&Bound. The main new ideas proposed in this article are that: (a) a concept called Weighted Consistency Nogood (WCN) allows to maintain consistency in DisWCSPs, (b) leading to an asynchronous equivalent of PFC-MRDAC. (c) The feedback that B&B or Adopt needs about low bounds can be extracted from such WCNs.

1. Introduction

Everybody has her problems. Private concerns can often be formulated in a general framework such as constraint satisfaction problems (where everything is modeled by either variables, values, or constraints) and then can be solved with any of the applicable CSP techniques. But often one has to find agreements with the other agents for a solution from the set of possible valuations that satisfy her subproblem. The framework modeling this kind of combinatorial problems is called Distributed Constraint Satisfaction.

In practice we should most often expect to meet an optimization problem rather than a satisfaction problem. Nevertheless the techniques developed for satisfaction problems have proved to be very useful when adapted to fit optimization problems (e.g. PFC-MRDAC [1]).

Distributed Weighted CSPs (DisWCSPs) is a general formalism that can model many negotiation problems and can quantify their privacy requirements. Here it is shown how a very general technique for Distributed CSPs, Replica-based Multiply Asynchronous Search (R-MAS), can be extended and applied to optimization in DisWCSPs.

Here is shown in detail how the technique of consistency maintenance in asynchronous search (DMAC-ABT) can be extended and hybridized with Adopt and Branch&Bound. DMAC-ABT is only a small part of R-MAS, but it is the single one needing modifications. All other techniques of R-MAS (reordering of ABTR, aggregation of AAS) apply almost unchanged to the extension. Namely small details change only in the local computation of AAS aggregates and these details are also discussed here.

2. Distributed Weighted CSPs

An extension of CSPs allowing for modeling some optimization functions is given by Weighted CSPs. When D is an ordered set, D=$\{D_1, D_2, ...D_m\}$, we will denote by $\vec{D}$ the set $D_1 \times D_2 \times \ldots \times D_m$. When $\vec{x}$ is a tuple of assignments to a tuple of variables $\vec{X}$, and $X' \subseteq X$, then $x|_{\vec{X'}}$ is the tuple obtained by projecting $\vec{x}$ unto $\vec{X'}$, namely the tuple of assignments from $\vec{x}$ for variables in $\vec{X'}$.

Definition 1 (WCSP) *A Weighted CSP is defined by a set of variables $X = \{x_1, x_2, ..., x_m\}$ taking values from a corresponding set of domains D=$\{D_1, D_2, ...D_m\}$ (x_i can take values from D_i) and a set of functions, $f_1, f_2, ..., f_i, ..., f_n$, of type $f_i : \vec{D} \rightarrow \mathbb{R}$ (not all variables are necessarily used in each function).*

The WCSP consists in finding $\operatorname{argmin}_{\vec{x} \in \vec{D}} \sum_{i=1}^{n} f_i(\vec{x})$.

A CSP is a particular type of WCSP where the functions f are predicates (aka *constraints*), namely functions with

results in the set $\{0, \infty\}$, meaning true respectively false. A solution is then defined as any $\vec{x}$ such that $\sum_{i=1}^{n} f_i(\vec{x}) = 0$. To model CSPs with WCSPs, infeasible tuples can be set to ∞, and feasible ones to 0.

Definition 2 (DisWCSP) *A Distributed Weighted CSP (DisWCSP) is defined by a set of agents $A_1, A_2, ..., A_n$, a set of variables $X = x_1, x_2, ..., x_m$ taking values from a corresponding set of domains $D=\{D_1, D_2, ...D_m\}$, and a set of functions $f_1, f_2, ...f_i, ..., f_n$, $f_i : \vec{D^i} \rightarrow \mathbb{R}$, D^i is the set of domains of the set of variables $X^i \subseteq X$. A_i is the only agent that knows f_i.*

The problem is to find $\underset{\vec{x} \in \vec{D}}{\operatorname{argmin}} \sum_{i=1}^{n} f_i(\vec{x}|_{\vec{X^i}})$ *where constraints for the domain of each existentially quantified variable x_i can be proposed by at least one agent.*

3. Replica-Based MAS

Multiply Asynchronous Search (MAS) is the algorithm that integrates asynchronous search, ABT, the reordering of ABTR, the consistency maintenance of DMAC, and the aggregations of AAS [4]. In Replica-based MAS (R-MAS) a set of distinct virtual agents (each enforcing another hierarchical level of abstraction, i.e. relaxations), replace together each physical agent's problem.

Agents exchange **ok?**, **add-link**, and **propagate** or **nogood** messages with aggregates, interests, respectively nogoods, according to the standard usage of these messages [4]. Abstracting from all details, the following concepts are used here.

Definition 3 *An* **aggregate** *(*assignment*) is a triplet $\langle x_j, s_j, h_j \rangle$ where x_j is a variable, s_j a set of values for x_j, $s_j \neq \emptyset$, and h_j a signature of the pair (x_j, s_j).*

Agents propose aggregates to restrict the possible values of variables in different contexts of the exploration of the search space. The signature helps to guarantee a correct message ordering. It determines if a given aggregate is more recent than another.

DMAC-ABT Maintaining Asynchronously Consistencies in asynchronous search (DMAC-ABT) is proposed among others in [4]. The algorithm consist in running ABT, on top of which distributed 'local' consistency achievement is enforced independently and concurrently for each subproblem generated by the the last proposed assignments of agents $A_i, i < k$. Each of the n subproblems is induced by taking for k a distinct value in $\{1..n\}$.

4. R-MAS for DisWCSPs

To use R-MAS with DisWCSPs, the idea is to model the value of a tuple in a proposal with a new variable.

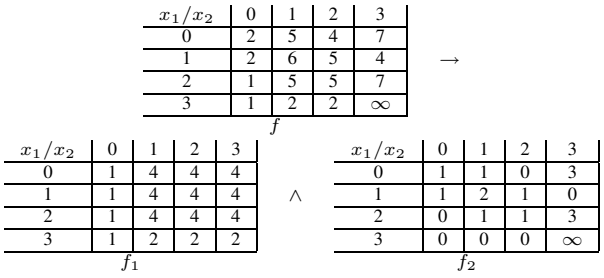

Figure 1. Example of constraint splitting.

Optional Aggregations Using aggregations is optional but interesting for privacy reasons. Aggregation is just a generalization of the case where a tuple is taken at a time. For correct evaluation of the value of a proposal in an aggregate, the proposed aggregates have to be built in such a way that the local cost is identical for all tuples of the known partial valuation (namely in the intersection of this proposal with all known valid proposals).

When constraints have tuples with very non-uniform values, one can still exploit wide aggregations by employing hierarchical abstractions. This can be done efficiently upon the technique of R-MAS, by allowing splitting of constraints (which is an abstraction technique) in such a way that all/several tuples in an abstract agent can be aggregated (see Figure 1). The only requirement is that the sum of the cost of a tuple in the obtained constraints equals the initial cost.

This splitting can be done in a greedy way, see the techniques in [4]. Alternatively, many clustering techniques can be straightforwardly used to get such splitting. This technique is optional and we will therefore not discuss it here to avoid over-burdening the reader. It is nevertheless introduced in notations to show that the description given extends with no modification to cases where splitting for aggregation will be used in the future.

Branch and Bound with cost variables Let us therefore introduce a new variable x_{c_i}, $x_{c_i} \geq 0$ for each agent A_i. These variables model the cost of the current proposal, the value of f_i, which should be the same for all valuations in the set of aggregates currently known by A_i. Since all agents are interested in the variables x_{c_i}, all the agents are in the *outgoing-links* of each agent A_i for the variable x_{c_i}. A_i proposes $x_{c_i}=\{k\}$ when he proposes an aggregate whose local cost is k (e.g. k=4 when the proposal is $(x_1 \in \{0..2\}, x_2 \in \{1..3\})$ for the first virtual agent obtained by splitting in Figure 1).

In Branch&Bound the idea is to discard search paths for which it is proven that any enclosed solution is more expensive than any already found solution. Any solution with

value C defines therefore a *nogood* (i.e. dynamically inferred constraint), $\sum_i x_{c_i} < C$, that is broadcast to all agents. It is known that $x_{c_i} \geq 0$ therefore each agent can enforce the weaker constraint: $\sum_{\text{known } x_{c_i}} x_{c_i} < C$.

No other modification is required and a new Branch and Bound algorithm is obtained. The last found solution is optimal. This algorithm is called R-MAS-BB-c1.

when received *(***solution**,C*)* **do**
add f(x) to the set of local constraints:

$$f(x) = \begin{cases} \infty & \text{if } \sum_{\text{known } x_{c_i}} x_{c_i} \geq C \\ 0 & \text{if } \sum_{\text{known } x_{c_i}} x_{c_i} < C \end{cases}$$

end do.
procedure solution-detected *(solution)* **do**
$C \leftarrow \sum_{(x_{c_i}, C_i, k_i) \in solution} C_i$;
broadcast (**solution**,C)
end do.

Algorithm 1: Procedure of A_i for receiving **solution** messages in R-MAS-BB-c1.All other procedures are inhereted from DMAC-ABT. The procedure *solution-detected* is run by whoever detects and builds the solution (e.g. broker). If each agent builds the solution separately then the message needs not be broadcasted but just delivered locally.

Remark 1 *With R-MAS-BB-c1, the value of a solution is given by the sum of the values assigned in it to all x_{c_i}.*

Solutions are detected in MAS according to the algorithm described in [4]. Each time that a solution is detected by the broker, a **solution** message is broadcasted to participants with the value C of the obtained solution. Algorithm 1 shows how the constraint $\sum_i x_{c_i} < C$ is added to each agent.

Example 1 *An* **ok?** *sent by the abstract agent of the first constraint obtained in Figure 1 can be:* **ok?***($\langle x_1, \{0\}, |1{:}0| \rangle \langle x_2, \{1..3\}, |1{:}0| \rangle \langle x_{c_1}, \{4\}, |1{:}0| \rangle$). This specifies that the R-MAS 'virtual' agent enforcing the constraint f_1 agrees to x_1=0, x_2 being any of the values $\{1..3\}$, and x_{c_1}=4.*

Proposition 1 *R-MAS-BB-c1 is correct, complet, terminates, and finds the optimal solution.*

Proof.*The proof is immediate from the correctness of R-MAS and by construction (introduction of Branch&Bound which is known to be correct).*

Cost of nogoods (WR-MAS) In the previous section it can be noticed that cost conflicts are only detected from partial valuations. A better idea has been introduced for centralized techniques in [1]. They explain how cost of subproblems can be computed by consistency propagation for estimating bounds earlier: *Use the cost of a constraint only once.*

In order to apply the previous techniques to R-MAS, we redefine the notion of consistency nogoods as follows.

Definition 4 (SRC) *A set of references to constraints (SRC), is a set of symbols (e.g. $\{C_{f_3}, C_{f_5}, C_{f_7}\}$), where C_{f_i} is a reference to the constraint, f_i, of the DisWCSP.*

Remark 2 *There is no need to attach a* reference to constraint *to hard constraints like $\sum_{known\ x_{c_i}} x_{c_i} < C$ since there is no problem in applying them redundantly: $\infty + \infty = \infty$.*

Definition 5 (Weighted Consistency nogood) *A weighted consistency* nogood *(WCN) for a level (i.e. search depth) k and a variable x has either the form $\langle srcs, c_1, c_2, V \cup (x \in l_x^k) \rangle$ or $\langle srcs, c_1, c_2, V \cup \neg(x \in s \setminus l_x^k) \rangle$.*

V is a set of assignments. Any assignment in V must have been proposed by A_k or its predecessors. l_x^k is a label, $l_x^k \neq \emptyset$. $srcs$ is a set of references to constraints while c_1 and c_2 are low bounds of the cost of the constraints referred by $srcs$ given V and values remaining, respectively values eliminated for x by l_x^k. s is the initial domain of x.

Remark 3 (Hard WCNs) *Most often c_2 will be ∞, therefore we will often use for WCNs the simplified notation $\langle srcs, c_1, V \cup (x \in l_x^k) \rangle$ that implies $c_2 = \infty$.*

Example 2 *Take as example the WCN $\langle \{C_{f_3}, C_{f_5}\}, 27, \infty, (\langle x_2, \{1..3\}, |1{:}0| \rangle \cup (x_4 \in \{3..5\})) \rangle$. This nogood states that as long as the assignment $\langle x_2, \{1..3\}, |1{:}0| \rangle$ is valid, the sum of the values due to the constraints referenced by C_{f_3}, C_{f_5} and unspecified hard nogoods is low bounded by 27 when $x_4 \in \{3..5\}$, respectively low bounded by $+\infty$ (i.e. infeasible) otherwise.*

The new concepts are the basis of a new family of asynchronous algorithms that extend R-MAS. We call the new family: Weighted R-MAS (WR-MAS).

Several WCNs can be stored by an agent for a variable at different depths in the search. A delicate problem is the combination of WCNs. Two consistency nogoods that can be combined in R-MAS can also be combined in WR-MAS in their weighted form.

Definition 6 *A* **weighted consistency nogood is valid** *only as long as all the aggregates involved in it are valid.*

Inference with weighted consistency nogoods

Proposition 2 *Any two weighted consistency nogoods, $\langle src_1, c_1, c_1', N_1 \cup x \in l_1 \rangle$ and $\langle src_2, c_2, c_2', N_2 \cup x \in l_2 \rangle$ where any aggregates in N_1 and N_2 for the same variable do not invalidate each other, can be combined into a new weighted consistency nogood. The obtained nogood is $\langle src, c, c', N \cup x \in l \rangle$ such that $src = src_1 \cup src_2$, $c = max(c_1, c_2)$, $c' = min(c_1', c_2')$, $l = l_1 \cap l_2$, and $N = N_1 \cup N_2$, N retaining only the strongest among two aggregates for the same variable.*

See several stronger inference rules in [3]. The operator combining two WCNs to the tightest WCN is denoted $\oplus$.

Example 3 *Consider first the general case:*
($\langle\{C_{f_1}, C_{f_2}\}, 27, \infty, x_1{\in}\{2..5\}\rangle \oplus \langle\{C_{f_2}, C_{f_3}\}, 15, 1000, x_2 \in \{4..7\}\rangle$*)*$\rightarrow \langle\{C_{f_1}, C_{f_2}, C_{f_3}\}, 27, 1000, x_2{\in}\{4, 5\}\rangle$*)*
For WCNs that have a single cost (see Remark 3):
($\langle\{C_{f_1}, C_{f_2}\}, 27, x_1{\in}\{2..5\}\rangle \oplus \langle\{C_{f_2}, C_{f_3}\}, 15, x_2 \in \{4..7\}\rangle$*)*$\rightarrow \langle\{C_{f_1}, C_{f_2}, C_{f_3}\}, 27, x_2{\in}\{4, 5\}\rangle$*)*

4.1. HOWTO infer WCNs

Generating/Strengthening WCNs Given a set N of valid assignments known by A_i and a set M of hard valid WCNs at search depth h, let T be the set of tuples allowed by them in the k-ary constraint f_i. We can infer k WCNs: $\langle srcs, c, V \cup (x_{ij} \in l^h_{ij})\rangle$, $j \in [1, k]$. Here:
a) $srcs$ is the union of the SRCs of the WCNs in M, and also contains the reference to f_i.
b) Let c_M be the maximum cost that can be obtained combining costs of WCNs. Costs are summed for WCNs having disjoint SRCs and the maximum is taken among costs of WCNs whose SRCs are not disjoint. Different order of applying these two operations lead to different results and backtracking is needed to search the order leading to the highest c_M. Let c_T be the minimum value that f_i attaches to a tuple in T. If the SRC of f_i is not in the SRCs found in M, then $c{=}c_T{+}c_M$, otherwise $c{=}max(c_T, c_M)$.
c) V is the union of all the assignments in M and N.
d) x_{ij} is the j^{th} variable involved in the k-ary constraint f_i.
e) l^h_{ij} is the label resulting at search depth h for x_{ij} after applying the proposals and WCNs in M and N.

Algorithm Filter in [3] shows in details the generation and propagation of WCNs.

Example 4 *We will consider the virtual agent* A_3 *enforcing the constraint* f_1 *in Figure 1. If* A_3 *knows an assignment* $x_2 \in \{1, 2\}$*|I:2| then it can infer the WCNs* $\langle\{C_{f_1}\}, 2, \langle x_2, \{1, 2\}, |I{:}2|\rangle \cup (x_2 \in \{1, 2\})\rangle$ *and* $\langle\{C_{f_1}\}, 2, \langle x_2, \{1, 2\}, |I{:}2|\rangle \cup (x_1 \in \{0..3\})\rangle$

Propagating WCNs A value v in the label of variable x is removed if the addition of an assignment $x \in \{v\}$ can lead to the inference of a WCN that together with the known constraints and assignments leads to an explicit nogood. A WCN $\langle srcs, c, L\rangle$ known by an agent A_i leads to an explicit nogood if A_i knows a constraint $\sum_{\text{known}} x_{c_i}\, x_{c_i} < C$, and it knows assignments for a set K of cost variables not in $srcs$ such that $sum_{(x_{c_k}, C_k) \in K} C_k + c \geq C$.

Example 5 *Consider that in Example 4,* A_1 *learns the nogood* $\sum_{known} x_{c_i}\, x_{c_i} < 7$ *from a message* **solution***(7) and receives the proposal* $\langle x_{c_3}, \{5\}, |2{:}3|\rangle$*. Each of the bounds* $b{\in}\{0, 1, 2\}$ *of* x_1 *fail successively if an assignment* $(x_1{\in}\{b\})$ *is added to the knowledge of* A_1*. Therefore* A_1 *obtains by propagation the WCN:* $\langle\{C_{f_1}\}, 2, \langle x_2, \{1, 2\}, |I{:}2|\rangle\langle x_{c_3}, \{5\}, |2{:}3|\rangle(x_1 \in \{3\})\rangle$

4.2. Data Structures for WR-MAS

The family of algorithms proposed here, Weighted Replica-based Multiply Asynchronous Search (WR-MAS), builds on R-MAS by replacing CNs with WCNs.

The following approaches are known for maintaining data structures with nogood-based consistency (considering that labels are treated as ranges):
a) DMAC0: Storing at most the last valid consistency nogood (CN) per variable.
b) DMAC1: Storing at most the last valid CN per variable per search depth (as in MHDC [4]).
c) DMAC2: Storing at most the last valid CN per variable per search depth per agent generating CNs (DMAC-ABT).
d) DMAC3: Storing at most the last valid CN per variable per search depth per agent generating CNs and per agent whose constraints are not involved in the CN (as in [4] for robustness in treating openness).

All of the previous four alternatives translate and work straightforwardly with WR-MAS, where one just uses WCNs instead CNs. The resulting techniques are therefore called: WDMAC0, WDMAC1, WDMAC2, WDMAC3. It is reasonable to expect that an important new alternative that becomes reasonable (for problems without openness or to allow the highest efficiency in combining WCNs) is to:
e) WDMAC4: Store at most the last valid WCN: i) per variable ($\times$ m variables), ii) per search depth ($\times a$ virtual agents), iii) per agent generating WCNs ($\times a$ virtual agents), iv) and per combination of involved SRCs (up to $k, k{\geq}2^c$, combinations may be used), i.e. $O(ma^2 2^c(md + c))$.

It can be noted that these requirements are exponential in the number of constraint references (number of weighted constraints). To meet space constraints, a fix subset of size k of combinations of constraint references has to be chosen. The corresponding space complexity becomes $O(ma^2 k(md + c))$ which is now polynomial.

4.3. Backtrack in Extended Branch and Bound

We have already presented almost all features on a new family of algorithms, Weighted Replica-based MAS (WR-MAS), and we have also seen in detail a particular algorithm of this family, namely DMAC-ABT. It remains us to explicit the way in which backtracking builds and sends explicit nogood messages. Weighted Replica-based MAS uses the previously mentioned form for WCNs. The cost C of any solu-

tion is broadcast under the form of a nogood, $\sum_i x_{c_i} < C$, as in R-MAS-BB-c1 (it could be similarly based on R-MAS-BB-c2 [4]). WR-MAS starts with all agents enforcing a constraint $\sum_i x_{c_i} < \infty$, and the constraints $x_{c_i} \geq 0$. They build and propose aggregates to all lower priority agents in corresponding outgoing-links. An agent builds a (hard) explicit nogood in any of the following cases (a more formal description appears in Algorithm **backtrack** in [3]):
a) It knows a constraint $\sum_i x_{c_i} < C$, and valid assignments to a set S of cost variables, such that $\sum_{i \in S} x_{c_i} \geq C$.
b) It knows a WCN $\langle src, c, M \rangle$, a constraint $\sum_i x_{c_i} < C$, and valid assignments to a set S of cost variables ($S \cap src = \emptyset$), such that $c + \sum_{i \in S} x_{c_i} \geq C$.
c) When it exhausts its search space and still cannot generate any new proposal, the agent generates an (eventually optimized) explicit nogood. The new explicit nogood is composed of valid received proposals by combining the nogoods entailed by the received valid proposals with the explicit nogoods previously received for its own proposals and that helped exhausting the current local search space.

Weighted Bound Consistency Arc consistency based on values has already been used many times with weighted CSPs. One of the most famous algorithms is given in [1]). In difference to existing versions we do not associate constraints to variables having to ensure that each constraint is counted only once by tricks in constraint representation, but rather we associate each weighted constraint with a constraint reference, keeping track of which constraints are involved in which costs. Based on (hard) WCNs, one can simply use ranges and Bound Consistency as in most discussed implementations of DMAC: As shown, a new weighted consistency nogood can be generated by proving that a certain bound of a variable leads to local cost that together with the view and nogoods involved in the computation lead to a conflict against a constraint $\sum_i x_{c_i} < C$.

Adopt±PFC-MRDAC In [5] we explained that there are two key elements that are required for improving efficiency for optimization with large problems: a) Limiting commitment. This consists in abandoning a branch if it is not promising. b) Using acceptable value ordering heuristics.

A solution is to abandon only when the heuristic has a higher confidence (e.g. the estimated cost of the current branch is (1+k) times more expensive than the estimation for the best alternative). To notice that the main theoretic result of A^* applies, namely: If the estimation of the cost of a path is either perfect or optimistic and if $k = 0$, then the first reached solution is the optimal one [2].

The algorithm obtained with this extension to DMAC-ABT is called Adopt-PFC-MRDAC (and the extension of the family of algorithms WR-MAS is called DWR-MAS.

5. Conclusions

At DCR 2001 we proposed an asynchronous Branch&Bound technique based on ABT/AAS. Some tests that we performed at that time have shown the technique to be prohibitivly expensive on a simple real-world problem. Recently, another technique, called Adopt, shows how A* value ordering heuristic can be introduced in ABT. While it is not yet known how the two existing asynchronous optimization techniques compare (namely Adopt vs ABT/AAS with Branch&Bound), here we have shown how these two techniques can be combined together and with local consistency maintenance.

The main new ideas proposed in this article are that:
1. Consistency achievement or maintenance in Weighted DisCSPs can be performed if the Consistency Nogood concept of DMAC-ABT is enriched to a more general concept: The Weighted Consistency Nogood (WCN). We prove rules of inference with WCNs.
2. An asynchronous equivalent of the best available centralized technique, PFC-MRDAC, is obtained by mixing the aforementioned consistency maintenance with Branch&Bound.
3. The feedback that Adopt needs about low bounds on constraints of successor agents, can be extracted using cost attached to labels in WCNs and detected by the previously mentioned 'local' consistency process.

In this paper we outlined the steps required for *asynchronizing* PFC-MRDAC for Distributed Weighted CSPs and we shown how consistency maintenance can be added to Adopt. More exactly PFC-MRDAC is obtained for certain synchronization and agent strategy in WR-MAS. DWR-MAS is nevertheless much more general and can lead to a very different behavior depending on network timing and agent strategies. Some interesting extensions (e.g. to arc consistency) are described in the extended version [3].

References

[1] J. Larrosa, P. Meseguer, and T. Schiex. Maintaining reversible DAC for Max-CSP. *AI*, 107:149–163, 1999.
[2] P. J. Modi, M. Tambe, W.-M. Shen, and M. Yokoo. A general-purpose asynchronous algorithm for distributed constraint optimization. In *Distributed Constraint Reasoning, Proc. of the AAMAS'02 Workshop*, Bologna, July 2002. AAMAS.
[3] M. Silaghi. Asynchronous PFC-MRDAC±Adopt —consistency-maintenance in adopt—. In *IJCAI-DCR*, 2003.
[4] M.-C. Silaghi. *Asynchronously Solving Distributed Problems with Privacy Requirements*. Phd thesis 2601, (EPFL), June 27, 2002. http://www.cs.fit.edu/~msilaghi/teza.
[5] M.-C. Silaghi, M. Calisti, and B. Faltings. On generalized english auctions. In *submitted to AAMAS-02*, page ID:393, September 2001. www.cs.fit.edu/~msilaghi/papers.