Improved Bounded Max-Sum for Distributed Constraint Optimization

Emma Rollon and Javier Larrosa

Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya, Spain.

Abstract. Bounded Max-Sum is a message-passing algorithm for solving Distributed Constraint Optimization Problems able to compute solutions with a guaranteed approximation ratio. Although its approximate solutions were empirically proved to be within a small percentage of the optimal solution on low and moderately dense problems, in this paper we show that its theoretical approximation ratio is overestimated, thus overshadowing its good performance. We propose a new algorithm, called Improved Bounded Max-Sum, whose approximate solutions are at least as good as the ones found by Bounded Max-Sum and with a tighter approximation ratio. Our empirical evaluation shows that the new approximation ratio is significantly tighter.

1 Introduction

Decentralised coordination techniques are a very important topic of research. A common approach is to cast the problem as a *multi-agent distributed constraint optimization problem* (DCOP), where the possible actions that agents can take are associated with *variables* and the utility for taking joint actions are encoded with (soft) *constraints* [8]. The set of constraints define a global utility function $F(x)$ to be optimized via decentralised coordination of the agents. In general, complete algorithms [6, 5, 7] (i.e. algorithms that find the true optimum) exhibit an exponentially increasing coordination overhead, which makes them useless in many practical situations.

Approximate algorithms constitute a very interesting alternative. They require little computation and communication at the cost of sacrificing optimality. There are several examples showing that they can provide solutions which are very close to optimality [3, 4]. However, this observation can only be verified on small toy instances, because it requires the computation of the true optimal to compare with, and it is not available in real-size real-world situations.

A significant breakthrough along this line of work was the Bounded Max-Sum algorithm (BMS) [8]. This algorithm comes with a guarantee approximation ratio $\widetilde{\rho}$, meaning that its approximate solution $\widetilde{\mathbf{x}}$ has a utility $F(\widetilde{\mathbf{x}})$ which is no more than a factor $\widetilde{\rho} \geq 1$ away from the optimum (i.e, $F(\widetilde{\mathbf{x}}) \leq F(\mathbf{x}^*) \leq \widetilde{\rho}F(\widetilde{\mathbf{x}})$). Clearly, large values of $\widetilde{\rho}$ reflect lack of confidence in the solution $\widetilde{\mathbf{x}}$. There are two possible reasons for a large $\widetilde{\rho}$: $i)$ the algorithm failed in finding a solution close to the optimal, $ii)$ the approximation ratio is not tight. Clearly, if we want $\widetilde{\rho}$ to be our measure of confidence about the quality of $\widetilde{\mathbf{x}}$, we want a tight $\widetilde{\rho}$ (i.e, $F(\mathbf{x}^*) \approx \widetilde{\rho}F(\widetilde{\mathbf{x}})$). Thus, the quality of the approximation ratio is a matter of the utmost importance.

In this paper we propose an improvement of BMS with approximation ratio ρ. We theoretically show that it is always better than the previous one (i.e., $\rho \leq \widetilde{\rho}$). Moreover, our experiments show that, in practice, ρ is much tighter than $\widetilde{\rho}$.

2 Preliminaries

In this Section we review the main elements to contextualize our work. Definitions and notation are borrowed almost directly from [8]. We urge the reader to visit that reference for more details and examples.

2.1 DCOP

A *Distributed Constraint Optimization Problem* (DCOP) is a quadruple $P = (\mathbf{A}, \mathbf{X}, \mathbf{D}, \mathbf{F})$, where $\mathbf{A} = \{\mathcal{A}_1, \ldots, \mathcal{A}_r\}$ is a set of agents, and $\mathbf{X} = \{x_1, \ldots, x_n\}$ and $\mathbf{D} = \{\mathbf{d}_1, \ldots, \mathbf{d}_n\}$ are variables and domains. $\mathbf{F} = \{f_1, \ldots, f_e\}$ is a set of cost functions. The objective function is,

$$F(x) = \sum_{j=1}^{e} f_j(x^j)$$

where $x^j \subseteq \mathbf{X}$ is the scope of f_j. A *solution* is a complete assignment $\mathbf{x}$. An *optimal solution* is a complete assignment $\mathbf{x}^*$ such that $\forall \mathbf{x},\ F(\mathbf{x}^*) \geq F(\mathbf{x})$. The usual task of interest is to find $\mathbf{x}^*$ through the coordination of the agents.

In the applications under consideration, the agents search for the optimum via decentralised coordination. We assume that each agent can control only its local variable(s) and has knowledge of, and can directly communicate with, a few neighboring agents. Two agents are neighbors if there is a relationship connecting variables and functions that the agents control.

The structure of a DCOP problem $P = (\mathbf{A}, \mathbf{X}, \mathbf{D}, \mathbf{F})$ can be transformed into a factor graph. A *factor graph* is a bipartite graph having a variable node for each variable $x_i \in \mathbf{X}$, a factor node for each local function $f_j \in \mathbf{F}$, and an edge connecting variable node x_i to factor node f_j if and only if x_i is an argument of f_j.

2.2 Max-Sum Algorithm

The *Max-Sum* algorithm [2, 1] is a message-passing algorithm for solving DCOP problems. It operates over a factor graph by sending functions (a.k.a., messages) along its edges. Edge (i, j) has associated two messages $q_{i\to j}$, from variable node x_i to function node f_j, and $r_{j\to i}$, from function node f_j to variable node x_i. These messages are defined as follows:

– **From variable to function:**

$$q_{i\to j}(x_i) = \alpha_{ij} + \sum_{k \in \mathcal{M}_i \setminus j} r_{k\to i}(x_i)$$

where $\mathcal{M}_i$ is a vector of function indexes, indicating which function nodes are connected to variable node x_i, and α_{ij} is a normalizing constant to prevent the messages from increasing endlessly in cyclic graphs.

– **From function to variable:**

$$r_{j\to i}(x_i) = \max_{x^j \setminus x_i} \{f_j(x^j) + \sum_{k \in \mathcal{N}_j \setminus i} q_{k\to j}(x_k)\}$$

where $\mathcal{N}_j$ is a vector of variable indexes, indicating which variable nodes are connected to function node f_j and $x^j \setminus x_i = \{x_k \mid k \in \mathcal{N}_j \setminus i\}$

Max-Sum is a distributed synchronous algorithm, since the agent controlling node i has to wait to receive messages from all its neighbors but j, to be able to compute (and send) its message to j. When the factor graph is cycle free, the algorithm is guaranteed to converge to the global optimal solution. Once the convergence is reached, each variable node can compute function,

$$z_i(x_i) = \sum_{k \in \mathcal{M}_i} r_{k\to i}(x_i)$$

The optimal solution is $\max_{x_i}\{z_i(x_i)\}$ and the optimal assignment $\mathbf{x}_i^* = \arg\max_{x_i}\{z_i(x_i)\}$. When the factor graph is cyclic, the algorithm may not converge to the optimum and only provides an approximation.

3 Bounded Max-Sum Algorithm

The *Bounded Max-Sum* algorithm (BMS) [8], is an approximation algorithm built on the Max-Sum algorithm. From a possibly cyclic problem P, the idea is to remove cycles in its factor graph by ignoring dependencies between functions and variables which have the least impact on the solution quality, producing a new acyclic problem $\widetilde{P}$. Then, Max-Sum is used to optimally solve $\widetilde{P}$ while simultaneously computing the approximation ratio $\widetilde{\rho}$. A more detailed description follows. For the sake of simplicity, we will restrict ourselves to the case of binary functions $f_j(x_i, x_k)$. The extension to general functions is direct. The algorithm works in three phases, each one implementable in a decentralised manner (see [8] for further details):

– **Relaxation Phase:** First, the algorithm weights each edge (i, j) of the original factor graph as,

$$w_{ij} = \max_{x_k}\{\max_{x_i} f_j(x_i, x_k) - \min_{x_i} f_j(x_i, x_k)\}$$

Then, it finds a maximum spanning tree T. Let W be the sum of weights of the removed edges (i.e, $W = \sum_{(i,j)\notin T} w_{ij}$). Next, the original problem P is transformed into an acyclic one $\widetilde{P}$ having the spanning tree T as factor graph. This is done as follows: for each edge (i, j) in the original graph that does not belong to the tree, the cost function $f_j(x_i, x_k)$ is transformed into another function $\widetilde{f_j}(x_k)$ defined as,

$$\widetilde{f_j}(x_k) = \min_{x_i} f_j(x_i, x_k)$$

Note that the objective function of $\widetilde{P}$ is

$$\widetilde{F}(x) = \sum_{(i,j),(k,j)\in T} f_j(x_i, x_k) + \sum_{(i,j)\notin T} \widetilde{f}_j(x_k)$$

- **Solving Phase:** BMS solves $\widetilde{P}$ with Max-Sum. Let $\widetilde{\mathbf{x}}$ be the solution of this problem. Since the factor graph of $\widetilde{P}$ is acyclic, $\widetilde{\mathbf{x}}$ is its optimal assignment.
- **Bounding Phase:** In [8], it is proved that,

$$F(\widetilde{\mathbf{x}}) \leq F(\mathbf{x}^*) \leq \widetilde{F}(\widetilde{\mathbf{x}}) + W$$

We can rewrite the previous upper bound expression as,

$$F(\mathbf{x}^*) \leq \frac{\widetilde{F}(\widetilde{\mathbf{x}}) + W}{F(\widetilde{\mathbf{x}})} F(\widetilde{\mathbf{x}})$$

Therefore, the algorithm computes $\widetilde{\rho} = \frac{\widetilde{F}(\widetilde{\mathbf{x}})+W}{F(\widetilde{\mathbf{x}})}$, which is a guarantee approximation ratio.

4 Improved BMS

4.1 Theoretical elements

Consider an edge (i, j) in the original factor graph that does not belong to the spanning tree. We define $\widehat{f}_j(x_k)$ as,

$$\widehat{f}_j(x_k) = \max_{x_i} f_j(x_i, x_k)$$

Let $\widehat{P}$ denote the problem containing the unmodified functions $f_j(x_i, x_k)$ (for $(i, j), (k, j) \in T$) and the $\widehat{f}_j(x_k)$ functions (for $(i, j) \notin T$). Note that $\widehat{P}$ and $\widetilde{P}$ have the same acyclic factor graph. Note as well that the objective function of $\widehat{P}$ is

$$\widehat{F}(x) = \sum_{(i,j),(k,j)\in T} f_j(x_i, x_k) + \sum_{(i,j)\notin T} \widehat{f}_j(x_k)$$

We can solve $\widehat{P}$ with Max-Sum. Let $\widehat{\mathbf{x}}$ be the optimal solution of this problem. It is obvious that $F(\widehat{\mathbf{x}})$ is a lower bound of $F(\mathbf{x}^*)$. Furthermore, as we prove next, $\widehat{F}(\widehat{\mathbf{x}})$ is an upper bound of $F(\mathbf{x}^*)$. Therefore, $\widehat{\rho} = \frac{\widehat{F}(\widehat{\mathbf{x}})}{F(\widehat{\mathbf{x}})}$ is a guarantee approximation ratio.

Theorem 1. $F(\mathbf{x}^*) \leq \widehat{F}(\widehat{\mathbf{x}})$.

Proof. By definition, $F(\mathbf{x}^*) = \sum_{(i,j),(k,j)\in T} f_j(\mathbf{x}_i^*, \mathbf{x}_k^*) + \sum_{(i,j)\notin T} f_j(\mathbf{x}_i^*, \mathbf{x}_k^*)$. Since for all f_j we have that $f_j(x_i, x_k) \leq \max_{x_i} f_j(x_i, x_k)$, then

$$F(\mathbf{x}^*) \leq \sum_{(i,j),(k,j)\in T} f_j(\mathbf{x}_i^*, \mathbf{x}_k^*) + \sum_{(i,j)\notin T} \max_{x_i} f_j(x_i, \mathbf{x}_k^*) = \widehat{F}(\mathbf{x}^*)$$

From the optimality of $\widehat{\mathbf{x}}$, we know that $\widehat{F}(\mathbf{x}^*) \leq \widehat{F}(\widehat{\mathbf{x}})$, which proves the theorem.

Next, we show that $\widehat{F}(\widehat{\mathbf{x}})$ is a tighter upper bound than $\widetilde{F}(\widetilde{\mathbf{x}}) + W$.

Theorem 2. $\widehat{F}(\widehat{\mathbf{x}}) \leq \widetilde{F}(\widetilde{\mathbf{x}}) + W$.

Proof. The proof is direct once it has been noted that for all $f_j(x_i, x_k)$,

$$\widehat{f}_j(x_k) \leq \widetilde{f}_j(x_k) + w_{ij}$$

which we prove next. By definition, the previous equation corresponds to,

$$\max_{x_i} f_j(x_i, x_k) \leq \min_{x_i} f_j(x_i, x_k) + \max_{x_k}\{\max_{x_i} f_j(x_i, x_k) - \min_{x_i} f_j(x_i, x_k)\}$$

which can be rewritten as,

$$\max_{x_i} f_j(x_i, x_k) - \min_{x_i} f_j(x_i, x_k) \leq \max_{x_k}\{\max_{x_i} f_j(x_i, x_k) - \min_{x_i} f_j(x_i, x_k)\}$$

which clearly holds.

We cannot establish any dominance relation between $\widetilde{\rho}$ and $\widehat{\rho}$ because there is no dominance between $F(\widetilde{\mathbf{x}})$ and $F(\widehat{\mathbf{x}})$. However, one way to circumvent this situation is to take $\rho = \frac{\widehat{F}(\widehat{\mathbf{x}})}{\max\{F(\widetilde{\mathbf{x}}), F(\widehat{\mathbf{x}})\}}$. The new ratio ρ dominates $\widetilde{\rho}$.

Theorem 3. $\rho \leq \widetilde{\rho}$.

Proof. Direct from Theorem 2 and the fact that $\max\{F(\widetilde{\mathbf{x}}), F(\widehat{\mathbf{x}})\} \geq F(\widetilde{\mathbf{x}})$.

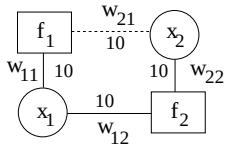
4.2 IBMS

Improved BMS (IMBS) works, as BMS, in three phases:

- **Relaxation Phase:** IBMS computes the spanning tree T and the relaxed problem $\widetilde{P}$ exactly as BMS does. Additionally, IBMS computes the relaxed problem $\widehat{P}$.
- **Solving Phase:** IBMS solves $\widetilde{P}$ and $\widehat{P}$ with Max-Sum. Let $\widetilde{\mathbf{x}}$ and $\widehat{\mathbf{x}}$ be the solutions of these problems. The agents will act according to the best solution ($\max\{F(\widetilde{\mathbf{x}}), F(\widehat{\mathbf{x}})\}$).
- **Bounding Phase:** IBMS computes the approximation ratio $\rho = \frac{\widehat{F}(\widehat{\mathbf{x}})}{\max\{F(\widetilde{\mathbf{x}}), F(\widehat{\mathbf{x}})\}}$.

The computation, storage and communication effort of IBMS is essentially twice that of BMS, because it requires solving two relaxed problems with Max-Sum. Given the low cost of BMS, doubling it seems acceptable. However, when it is not the case, one can always run a weaker version of IBMS ignoring $\widetilde{P}$. This weaker version will be exactly as costly as BMS. Its disadvantage is that $\widehat{\mathbf{x}}$ is not guaranteed to be better than $\widetilde{\mathbf{x}}$. In fact, our experiments show that there is no clear winner among them. Interestingly, the approximation ratio of the weaker version $\widehat{\rho}$ is systematically better than the approximation ratio of BMS $\widetilde{\rho}$.

Example 1. Consider the problem P given in Figure 1 with two variables $\{x_1, x_2\}$ and two functions $\{f_1, f_2\}$. The spanning tree of its factor graph is given with solid lines (i.e., edge (x_2, f_1) has been removed, shown as a dashed line). Thus, $W = 10$. Functions $\widetilde{f}_1$ and $\widehat{f}_1$ in $\widetilde{P}$ and $\widehat{P}$, respectively, are given in the figure. Max-Sum finds assignments $\widetilde{\mathbf{x}} = \widehat{\mathbf{x}} = (x_1 = a, x_2 = a)$, with utility $\widetilde{F}(\widetilde{\mathbf{x}}) = \widehat{F}(\widehat{\mathbf{x}}) = 20$. Their evaluation on the original problem P is $F(\widetilde{\mathbf{x}}) = F(\widehat{\mathbf{x}}) = 20$. The approximation ratios are $\widetilde{\rho} = 1.5$, $\widehat{\rho} = 1$, and $\rho = 1$.



x_1	x_2	f_1
a	a	10
a	b	10
b	a	0
b	b	10

x_1	x_2	f_2
a	a	10
a	b	0
b	a	0
b	b	10

x_1	$\widetilde{f}_1$
a	10
b	0

x_1	$\widehat{f}_1$
a	10
b	10

Fig. 1. Example of a factor graph containing cycles and a spanning tree formed by removing the edge between variable node x_2 and function node f_1.

5 Empirical Evaluation

The purpose of the experiments is to evaluate the improvement of our upper bound $\widehat{F}(\widehat{\mathbf{x}})$ and approximation ratios ρ and $\widehat{\rho}$ over the BMS upper bound $\widetilde{F}(\widetilde{\mathbf{x}}) + W$ and approximation ratio $\widetilde{\rho}$, respectively. We consider the same set of problems from the ADOPT repository[1] used in [8]. These problems represent graph coloring problems with two different link densities (i.e., the average connection per agent) and different number of nodes. Each agent controls one node (i.e., variable), with domain $|\mathbf{d}_i| = 3$, and each edge of the graph represents a pairwise constraint between two agents. Each edge is associated with a random payoff matrix, specifying the payoff that both agents will obtain for every possible combination of their variables' assignments. Each entry of the payoff matrix is a real number sampled from two different distributions: a gamma distribution with $\alpha = 9$ and $\beta = 2$, and a uniform distribution with range $(0, 1)$. For each configuration, we report average values over 25 repetitions. For the sake of comparison, we compute the optimal utility by a complete centralized algorithm, although this value can only be computed up to 12 agents by a complete decentralized algorithm, as shown in [8].

Figure 2 (first and second rows) shows the upper and lower bound obtained by IBMS (i.e., $\widehat{F}(\widehat{\mathbf{x}})$ and $\max\{F(\widehat{\mathbf{x}}), F(\widetilde{\mathbf{x}})\}$, respectively) and BMS (i.e., $\widetilde{F}(\widetilde{\mathbf{x}})$ and $F(\widetilde{\mathbf{x}})$, respectively), along with the optimal utility (i.e., $F(\mathbf{x}^*)$), for the different link densities and payoff distributions. The behavior of both algorithms is very similar across all link densities and payoff distributions. IBMS always computes an upper bound tighter than the one computed by BMS. The improvement is slightly better for the uniform distribution. The lower bounds computed by both algorithms are very close, although IBMS lower bound is slightly better.

Figure 2 (bottom row) shows a detail on the lower bounds $F(\widehat{\mathbf{x}})$ and $F(\widetilde{\mathbf{x}})$ obtained on each instance of a given parameter configuration. Since the behavior across all number of agents, link densities and payoff distributions is very similar, we only report results on instances with 25 agents and gamma distribution. Both lower bounds are very close, and none of them is consistently better than the other.

Figure 3 shows the percentage of improvement of the approximation ratio of IBMS ρ and the weaker version of IBMS $\widehat{\rho}$ over the approximation ratio of MBS $\widetilde{\rho}$ (left y-axe). The figure also reports the percentage of deterioration of the approximation ratio of the 7-size-bounded-distance criteria introduced in [9] according to the minimum maximum

[1] http://teamcore.usc.edu/dcop

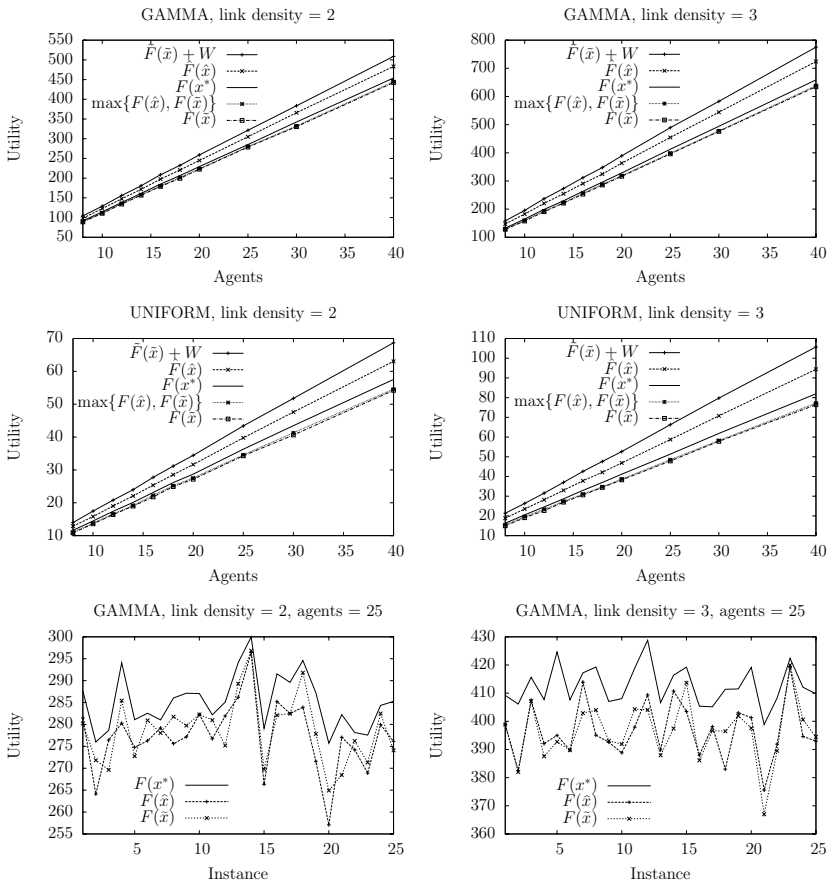


Fig. 2. First and second row, bounds obtained by algorithms IBMS and BMS varying the number of agents; third row, lower bound detail for instances with 25 agents and gamma distribution.

reward bound ($S7_r$) and the minimum fraction bound ($S7_f$) presented in [10] over the approximation ratio of MBS $\widetilde{\rho}$ (right y-axe). Since the relation between the optimal solution of the problem $F(\mathbf{x}^*)$ and an approximation ratio ρ of a given solution $\mathbf{x}$ is $1 \leq \frac{F(\mathbf{x}^*)}{F(\mathbf{x})} \leq \rho$, we compute the improvement of an approximation ratio ρ over $\widetilde{\rho}$ as,

$$\frac{(\widetilde{\rho} - 1) - (\rho - 1)}{\widetilde{\rho} - 1} * 100$$

The improvement of ρ is always higher than 37%, and up to almost 50%. Its mean improvement for the gamma and uniform distributions is higher than 40% and 45%, respectively. The improvement of $\widehat{\rho}$ is always higher than 32%, and up to almost 46%. Its mean improvement for the gamma and uniform distributions is higher than 35% and 37%, respectively. Therefore, both IBMS and its weaker version always significantly

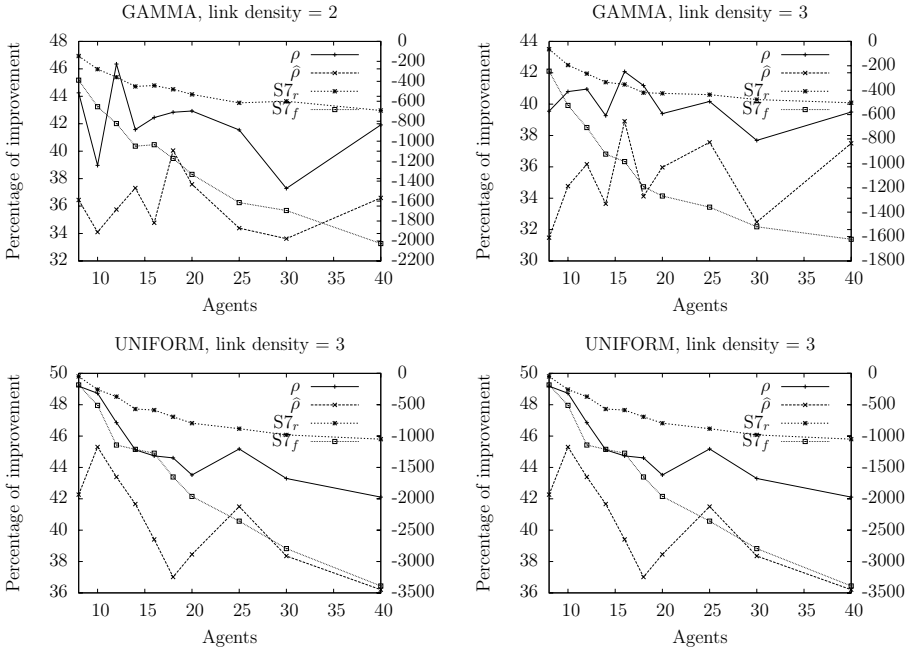


Fig. 3. Percentage of improvement of the approximation ratio of IBMS ρ and weaker version of IBMS $\widehat{\rho}$ (left y-axe), and percentage of decrease of the 7-size-bounded-distance criteria using the the minimum maximum reward bound ($S7_r$) and the minimum fraction bound ($S7_f$) (right y-axe) over the approximation ratio of BMS $\widetilde{\rho}$.

outperforms BMS. Recall that the weaker version of IBMS has the same communication demands as BMS. Both approximation ratios $S7_r$ and $S7_f$ are worse than the approximation ratio of BMS $\widetilde{\rho}$ (the percentage is always negative). Their quality decreases as the number of agents increases for both distributions.

6 Conclusions

In this paper we introduced a new algorithm, called Improved Bounded Max-Sum (IBMS), based on the Bounded Max-Sum algorithm. We theoretically proved that its approximation ratio is always better than the previous one, at the only cost of doubling the communication requirements. We also introduced a weaker version of IBMS having the same communication demands as Bounded Max-Sum. Our experiments show that the approximation ratio of both algorithms is significantly tighter.

Acknowledgements. The authors are thankful to the authors of [8, 9] for providing us with their implementations, and to the authors of [8] for sharing their benchmarks with us. This work was supported by project TIN2009-13591-C02-01.

References

1. S. M. Aji and R. J. McEliece. The generalized distributive law. *IEEE Transactions on Information Theory*, 46(2):325–343, 2000.
2. A. Farinelli, A. Rogers, A. Petcu, and N. R. Jennings. Decentralised coordination of low-power embedded devices using the max-sum algorithm. In *AAMAS*, pages 639–646, 2008.
3. S. Fitzpatrick and L. Meetrens. Distributed coordination through anarchic optimization. In *Distributed Sensor Networks A multiagent perspective*, pages 257–293. Kluwer Academic, 2003.
4. R. J. Maheswaran, J. Pearce, and M. Tambe. A family of graphical-game-based algorithms for distributed constraint optimization problems. In *Coordination of Large-Scale Multiagent Systems*, pages 127–146. Springer-Verlag, 2005.
5. R. Mailler and V. R. Lesser. Solving distributed constraint optimization problems using cooperative mediation. In *AAMAS*, pages 438–445, 2004.
6. P. J. Modi, W. M. Shen, M. Tambe, and M. Yokoo. Adopt: asynchronous distributed constraint optimization with quality guarantees. *Artif. Intell.*, 161(1-2):149–180, 2005.
7. A. Petcu and B. Faltings. A scalable method for multiagent constraint optimization. In *IJCAI*, pages 266–271, 2005.
8. A. Rogers, A. Farinelli, R. Stranders, and N. R. Jennings. Bounded approximate decentralised coordination via the max-sum algorithm. *Artif. Intell.*, 175(2):730–759, 2011.
9. M. Vinyals, E. Shieh, J. Cerquides, J. A. Rodriguez-Aguilar, Z. Yin, M. Tambe, and E. Bowring. Quality guarantees for region optimal dcop algorithms. In *AAMAS*, pages 133–140, 2011.
10. M. Vinyals, E. Shieh, J. Cerquides, J. A. Rodriguez-Aguilar, Z. Yin, M. Tambe, and E. Bowring. Reward-based region optimal quality guarantees. In *OPTMAS Workshop*, 2011.