Risk-Neutral Bounded Max-Sum for Distributed Constraint Optimization

Javier Larrosa
Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya, Spain.
larrosa@lsi.upc.edu

Emma Rollon
Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya, Spain.
erollon@lsi.upc.edu

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 moderate dense problems, in this paper we show that a simple modification systematically provides even better solutions. This is especially relevant in critical applications (e.g. disaster response scenarios) where the accuracy of solutions is of vital importance.

1. INTRODUCTION

Recently, significant research effort has sought to apply coordination techniques to control physical devices that are able to acquire information from the environment. In this context, *Decentralized 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* [11]. The set of constraints define a global utility function $F(x)$ to be optimized via decentralized coordination of the agents. In general, complete algorithms [7, 6, 9] (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 of very good quality [3, 5].

A significant breakthrough along this line of work was the *Bounded Max-Sum algorithm* (BMS) [11]. This algorithm provides, with very little coordination, solutions that are very close to the optimum on low and moderate dense problems. Thus, BMS is especially suitable for critical applications like *disaster response*, where it is critical to obtain almost instantly, very accurate solutions [13, 14, 10]. In this problems there are multiple mobile sensors that gather information in crisis situations. These mobile sensors could be autonomous ground robots or unmanned aerial vehicles. In either case, while patrolling through the disaster area, these sensors need to keep track of the continuously changing state of spatial phenomena, such as temperature or the concentration of potentially toxic chemicals.

Arguably, the most interesting feature of BMS is that it comes with a *guarantee approximation ratio*, meaning that its approximate solution has a utility which is no more than a factor away from the optimum. Clearly, large values of the ratio reflect lack of confidence in the solution. There are two possible reasons for a large ratio: i) the algorithm failed in finding a solution close to the optimal, ii) the approximation ratio is not tight. In a recent paper, Rollon and Larrosa [12] addressed the second issue and proposed an improved BMS (IBMS) with a much tighter upper bound. In this paper, we consider the first issue and propose a modification of BMS (that we call RN-BMS) with which tighter lower bounds are obtained.

The three algorithms under consideration (BMS, IBMS and our RN-BMS) relax the problem transforming some n-ary utility functions into unary. As we show, the unary function of BMS promotes a *risk-averse* behaviour of the agent, when *guessing* the information lost throughout the relaxation. IBMS promotes a *risk-loving* behaviour. Our algorithm, RN-BMS, allows a *risk-neutral* behaviour. Our experiments, show that RN-BMS systematically obtains better solutions.

2. PRELIMINARIES

In this Section we review the main elements to contextualize our work. Definitions and notation are borrowed almost directly from [11]. We urge the reader to visit that reference for more details and examples.

2.1 DCOP

A *Distributed Constraint Optimization Problem* (DCOP) is a tuple $P = (\mathbf{A}, \mathbf{X}, \mathbf{D}, \mathbf{F})$, 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 decentralized coordination. We assume that each

SAC'13 March 18-22, 2013, Coimbra, Portugal.

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. PREVIOUS BOUNDED MAX-SUM ALGORITHMS

The *Bounded Max-Sum* algorithms, BMS [11] and weak IBMS and IBMS [12], are approximation algorithms 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. Then, Max-Sum is used to optimally solve the acyclic problem while simultaneously computing the approximation ratio. IBMS is the combination of weak IBMS and BMS. 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 algorithms works in three phases, each one implementable in a decentralized manner (see [11] for further details). In this section, we will restrict our attention to the first and second phases. The third phase of BMS, weak IBMS and our RN-BMS will be discussed in a separate section (see Section 5).

3.1 Bounded Max-Sum Algorithm (BMS)

The BMS algorithm [11] works as follows:

- **Relaxation Phase:** First, the algorithm assigns a weight w_{ij} to each edge (i, j) of the original factor graph measuring the impact that the factor may have in the optimal solution. Then, it finds a maximum spanning tree T with respect to the weights. 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. Obviously, $F(\widetilde{\mathbf{x}})$ is a lower bound of the optimal solution ($F(\widetilde{\mathbf{x}}) \leq F(\mathbf{x}^*)$).

3.2 Weak Improved BMS (weak IBMS)

The weak IBMS algorithm [12], which allows the computation of better upper bounds (see Section 5), works as follows:

- **Relaxation Phase:** The original problem P is transformed into an acyclic one $\widehat{P}$. For the transformation, the cost function $f_j(x_i, x_k)$ of each edge (i, j) in the original graph that does not belong to the tree is transformed into another function $\widehat{f}_j(x_k)$ defined as,

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

  Thus, the objective function of $\widehat{P}$ is

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

- **Solving Phase:** weak IBMS solves $\widehat{P}$ with Max-Sum. Let $\widehat{\mathbf{x}}$ be the solution of this problem. As with BMS, it is obvious that $F(\widehat{\mathbf{x}})$ is a lower bound of the optimal solution ($F(\widehat{\mathbf{x}}) \leq F(\mathbf{x}^*)$).

4. IMPROVING THE LOWER BOUND OF BMS AND WEAK IBMS

Consider a problem where variables have three domain values $\{a, b, c\}$. Let $f_j(x_i, x_k)$ be the cost function depicted in Figure 1 (first table). Let us suppose that we apply one of the previous algorithms and that $f_j(x_i, x_k)$ is one of the cost functions that needs to be removed. Both algorithms, BMS and weak IBMS, will replace the binary function $f_j(x_i, x_k)$ by a unary one. Figure 1 (second table) shows function $\widetilde{f}_j(x_k)$ computed by BMS and Figure 1 (third

table) shows $\widehat{f}_j(x_k)$ computed by weak IBMS. Clearly, that means that the new, relaxed, problem will lose the *connection* between x_i and x_k. In other words, the new problem will have no further knowledge about how good the different combinations of values are. In turn, there will be a new unary function. Since, this unary function assigns utilities to the different values, it will increase or decrease the likeliness of the different values of x_k of being in the solution.

As we show next, the idea behind RN-BMS is to compute a unary function that promotes the behaviour of risk-neutral agent. For the sake of the discussion, let us look at Figure 1 (first table) and consider the different values for x_k.

- Value a is the best option in a *worst-case* scenario. It will provide an utility of at least 10, which is more that what the other two values can guarantee. Thus, it would be the choice of a *risk-averse agent*. Figure 1 (second table) shows that this is the rational behind BMS, since $\widetilde{f}_j(x_k)$ assigns the highest utility to a.
- Value b is the best option in a *best-case* scenario. It may provide the highest utility (1100), which is more that what the other two values can offer. But this will only happen if value a is also the choice for variable x_i, and the other two options for x_i are really bad. Thus, it would be the choice of a *risk-loving* agent. Figure 1 (third table) shows that this is the rational behind IBMS, since $\widehat{f}_j(x_k)$ assigns the highest utility to b.
- However, value c seems the most neutral option, since there are 2 out of 3 very good options. This would be the choice of a *risk-neutral agent*. This behaviour is encoded if the unary function is computed taking (possibly, weighted) averages. This is the rational of our new algorithm RN-BMS.

The *Risk-Neutral Bounded Max-Sum* algorithm (RN-BMS) works as follows.

- **Relaxation Phase:** RN-BMS assumes that, each agent has belief functions on the neighbour variables. Consider the agent controlling variable x_k. Let x_i be a neighbour variable and $f_j(x_i, x_k)$ be the corresponding utility function. The agent has function $b_j(x_i)$ which measures, for each value in the domain of x_i, its belief on how likely it is that the value will be in an optimal solution. Then, if function $f_j(x_i, x_k)$ is to be eliminated, it is replaced by a unary function $f_j^N(x_k)$ defined as,

$$f_j^N(x_k) = \sum_{x_i} f_j(x_i, x_k) b_j(x_i)$$

  Thus, the objective function of P^N is

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

  For the sake of illustration and further experimentation, we consider the simplest case, in which beliefs are probabilities, and the agent considers all values equally probable. Under this assumption, the unary function contains the *average* utility over the binary extensions. Figure 1 (fourth table) shows the resulting unary function in our running example.

  Clearly, our framework allows more sophisticated believe functions. For instance, the agents could *believe* that some values are more relevant than others based on their previous experience.

- **Solving Phase:** RN-BMS solves P^N with Max-Sum. Let $\mathbf{x}^N$ be the solution of this problem. As in the previous cases, it is obvious that $\mathbf{x}^N$ is a lower bound of the optimal solution ($F(\mathbf{x}^N) \leq F(\mathbf{x}^*)$).

5. APPROXIMATION RATIO

An approximate algorithm providing an approximate solution $\mathbf{x}$ and a *guarantee approximation ratio* $\rho \geq 1$, means that its solution has a utility $F(\mathbf{x})$ which is no more than a factor ρ away from the optimum (i.e, $F(\mathbf{x}) \leq F(\mathbf{x}^*) \leq \rho F(\mathbf{x})$). This approximation ratio is computed in the third phase, called *bounding phase*, of the bounded max-sum algorithms as follows.

- **BMS.** Let $w_{ij} = \max_{x_k}\{\max_{x_i} f_j(x_i, x_k) - \min_{x_i} f_j(x_i, x_k)\}$ and $W = \sum_{(i,j) \notin T} w_{ij}$. In [11], it is proved that,

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

  We can rewrite the previous bounding expression as,

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

  Therefore, $\widetilde{\rho} = \frac{\widetilde{F}(\widetilde{\mathbf{x}}) + W}{F(\widetilde{\mathbf{x}})}$ is a guarantee approximation ratio for BMS.

- **Weak IBMS.** In [12], it is proved that,

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

  Therefore, $\widehat{\rho} = \frac{\widehat{F}(\widehat{\mathbf{x}})}{F(\widehat{\mathbf{x}})}$ is a guarantee approximation ratio for weak IBMS.

- **RN-BMS.** Let

$$w_{ij}^N = \max_{x_k}\{\max_{x_i} f_j(x_i, x_k) - \sum_{x_i} f_j(x_i, x_k) b_j(x_i)\}$$

  and $W^N = \sum_{(i,j) \notin T} w_{ij}^N$. Following a reasoning similar to the BMS case, one can see that

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

  Therefore, $\rho^N = \frac{F^N(\mathbf{x}^N) + W^N}{F(\mathbf{x}^N)}$ is a guaranteed approximation ratio for RN-BMS.

In [12], it is proved that,

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

Since IBMS is the combination of BMS and weak IBMS, its guarantee approximation ratio is $\rho^I = \frac{\widehat{F}(\widehat{\mathbf{x}})}{\max\{F(\widehat{\mathbf{x}}), F(\widehat{\mathbf{x}})\}}$.

Following similar reasonings, one can see that

$$\widehat{F}(\widehat{\mathbf{x}}) \leq F^N(\mathbf{x}^N) + W^N$$

However, there is no theoretical dominance between the upper bounds obtained by BMS and RN-BMS, nor by their lower bounds. Thus, we cannot establish any dominance between the three approximation ratios $\widetilde{\rho}$, $\widehat{\rho}$, and ρ^N. However, we can establish a ratio

$$\rho = \frac{\widehat{F}(\widehat{\mathbf{x}})}{\max\{F(\widetilde{\mathbf{x}}), F(\widehat{\mathbf{x}}), F(\mathbf{x}^N)\}}$$

which dominates all of them. However, it requires the execution of the three algorithms.

x_k	x_i	f_j
a	a	10
a	b	11
a	c	12
b	a	1100
b	b	0
b	c	0
c	a	1000
c	b	1000
c	c	0

x_k	$\widetilde{f}_j$
a	10
b	0
c	0

x_k	$\widehat{f}_j$
a	12
b	1100
c	1000

x_k	f_j^N
a	11
b	366
c	666

Figure 1: Example of a binary utility function and its unary relaxation as computed by BMS, IBMS and RN-BMS

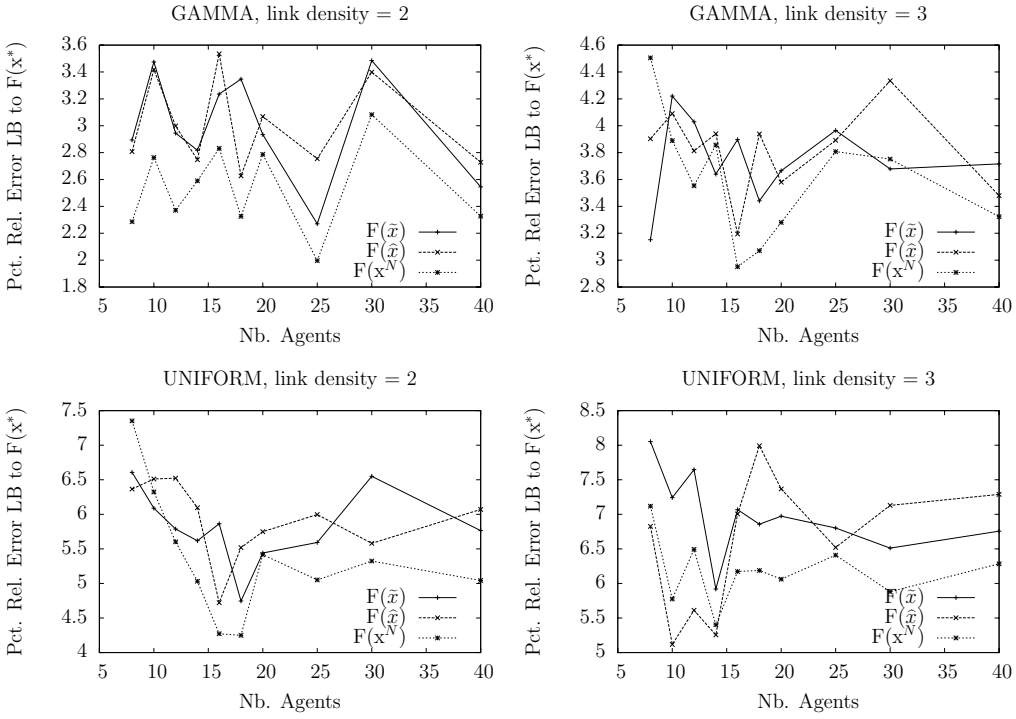

Figure 2: Percentage relative error of the lower bound obtained by BMS ($F(\widetilde{\mathbf{x}})$), weak IBMS ($F(\widehat{\mathbf{x}})$), and RN-BMS ($F(\mathbf{x}^N)$) to the optimum $F(\mathbf{x}^*)$.

6. EMPIRICAL EVALUATION

The main purpose of the experiments is to evaluate the improvement of the lower bound $F(\mathbf{x}^N)$ with respect to the BMS and weak IBMS lower bounds $F(\widetilde{\mathbf{x}})$ and $F(\widehat{\mathbf{x}})$, respectively. We consider the same set of problems from the ADOPT repository[1] used in [11]. 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.

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

For the sake of comparison, we compute the optimal utility with a complete centralized algorithm, although this value can only be computed up to 12 agents with a complete decentralized algorithm, as shown in [11].

Figure 2 shows the percentage relative error of the lower bound obtained by BMS ($F(\widetilde{\mathbf{x}})$), weak IBMS ($F(\widehat{\mathbf{x}})$), and RN-BMS ($F(x^N)$) over the optimum $F(\mathbf{x}^*)$ for the different link densities and payoff distributions. The percentage relative error of a given lower bound LB to the optimum $F(\mathbf{x}^*)$ is

$$\frac{F(\mathbf{x}^*) - LB}{F(\mathbf{x}^*)} * 100$$

The improvement of RN-BMS is very relevant. On the one hand, recall that these class of algorithms are being developed for applications in which the accuracy of the solution is extremely important. On the other hand, although BMS and weak IBMS are already very accurate, the approximate solution found by RN-BMS is even tighter.

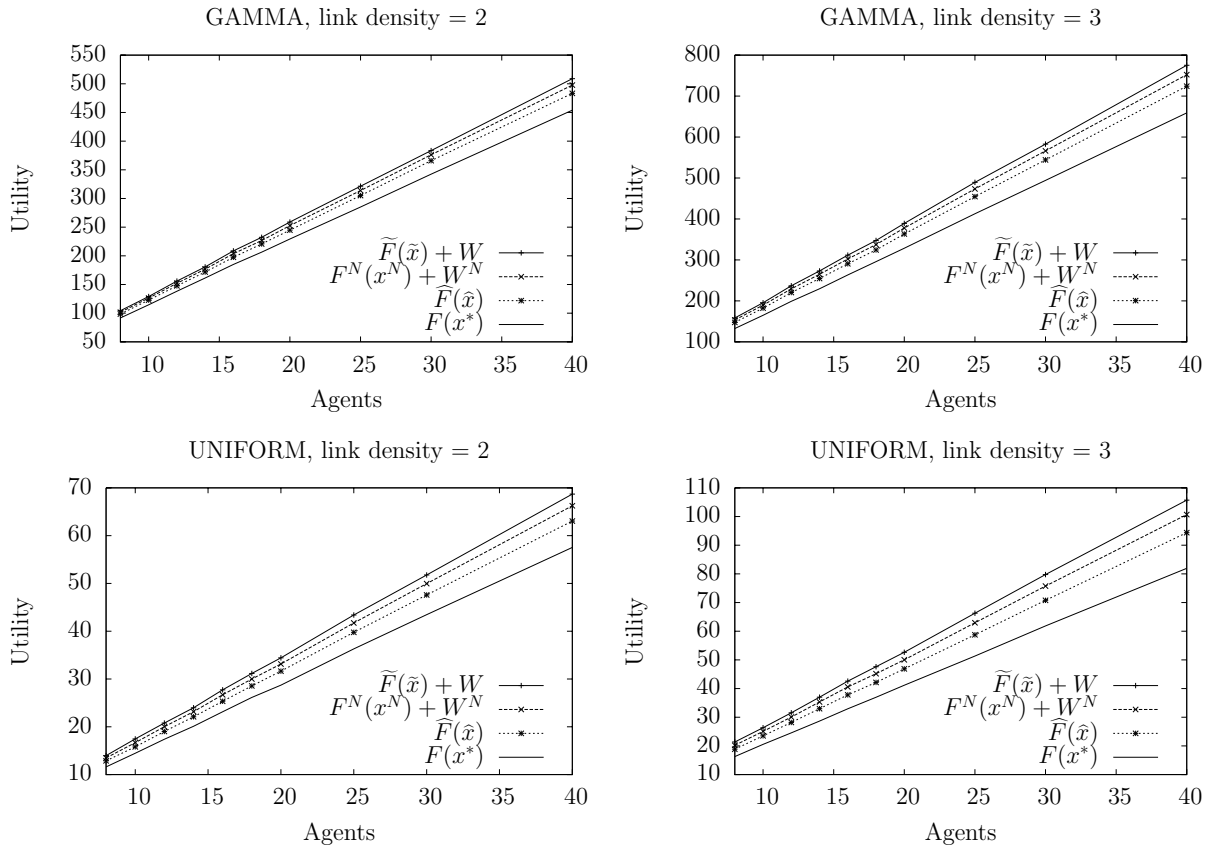

Figure 3: Upper bounds obtained by BMS ($\widetilde{F}(\widetilde{\mathbf{x}}) + W$), (weak) IBMS($\widehat{F}(\widehat{\mathbf{x}})$), and RN-BMS ($F^N(\mathbf{x}^N) + W^N$) along with the optimum ($F(\mathbf{x}^*)$).

Figure 3 shows the upper bound obtained by BMS ($\widetilde{F}(\widetilde{\mathbf{x}}) + W$), (weak) IBMS($\widehat{F}(\widehat{\mathbf{x}})$), and RN-BMS ($F^N(x^N) + W^N$). For comparison, the figure also reports the optimum ($F(\mathbf{x}^*)$). The behaviour of all algorithms is very similar across all link densities and payoff distributions. As theoretically proved, weak IBMS always computes the tightest upper bound. Note that, although theoretically incomparable, RN-BMS is always superior to BMS across all instances.

Figure 4 shows the approximation ratios obtained by BMS ($\widetilde{\rho}$), weak IBMS ($\widehat{\rho}$) and RN-BMS (ρ^N). The figure also reports the best approximation ratio (ρ), that is, using the best upper bound (given by weak IBMS) and the best lower bound. As discussed in [12], computing such ratio requires to linearly increase the coordination work. Weak IBMS is superior to the other two single algorithms, mainly due to its upper bound accuracy. Clearly, the combination of the three algorithms computes a good approximate solution with very high confidence.

7. RELATED WORK

There are other two incomplete algorithms that can provide guarantees on the worst-case solution quality of their solutions at design time: *k-optimality* [8] and *t-optimality* [4]. The idea of these algorithms is to form coalitions of agents and to find the local optima solutions for all agents within the coalitions. This local optima is guaranteed to be within a predefined distance from the global optimal solution.

Very recently, [15] proposed a framework were different coalition-based local optimality schemes can be described and defined a new criteria called *s-size bounded optimality*. The complexity of these algorithms depend on the number of coalitions and their size. Therefore, in practice, these algorithms are used with relatively small values of their control parameter.

In [11], it was shown that k-optimality provided significantly worst quality guarantees than BMS for different values of k. In [12], it was shown that s-size-bounded-distance provided worse approximation ratios than IBMS and BMS, even using the improved minimum maximum reward and minimum fraction bounds proposed in [16] and a relatively high value of the parameter s.

8. CONCLUSIONS AND FUTURE WORK

In this paper we introduced a new algorithm, called Risk-Neutral Bounded Max-Sum (RN-BMS), based on the Bounded Max-Sum algorithm. We show that it implements a Risk-Neutral approach to Bounded Max-Sum, in contrast to BMS and IBMS which implement risk-loving and risk-averse approaches, respectively. Our experiments show that RN-BMS systematically provides better solutions.

Our experiments assume that agents have no rich knowledge about which values are more likely to be in the optimal solution. However, our framework allows more elaborated implementation. Its impact remains as future work.

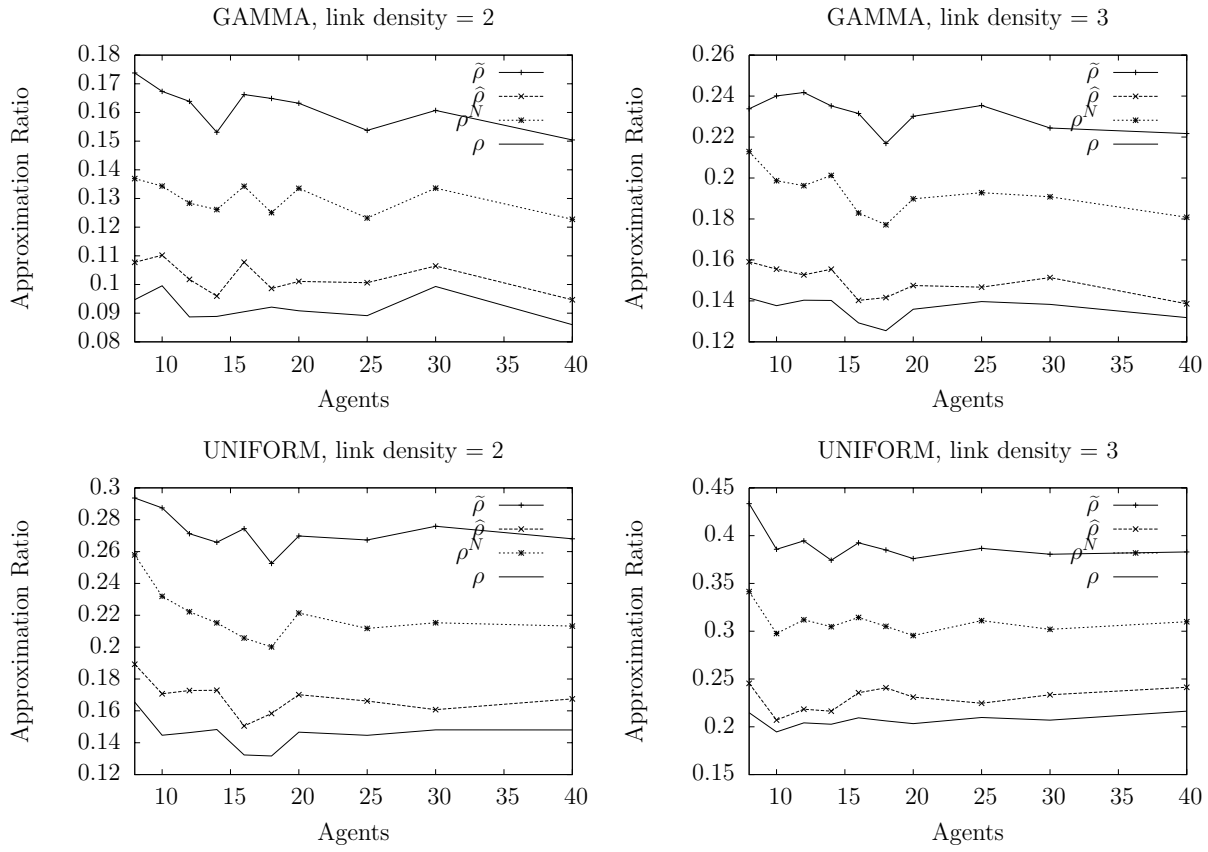

Figure 4: Approximation ratio obtained by BMS ($\widetilde{\rho}$), weak IBMS ($\widehat{\rho}$), RN-BMS (ρ^N), and the best among them (ρ).

9. 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] C. Kiekintveld, Z. Yin, A. Kumar, and M. Tambe. Asynchronous algorithms for approximate distributed constraint optimization with quality bounds. In *AAMAS*, pages 133–140, 2010.

[5] 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.

[6] R. Mailler and V. R. Lesser. Solving distributed constraint optimization problems using cooperative mediation. In *AAMAS*, pages 438–445, 2004.

[7] 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.

[8] J. P. Pearce and M. Tambe. Quality guarantees on k-optimal solutions for distributed constraint optimization problems. In *IJCAI*, pages 1446–1451, 2007.

[9] A. Petcu and B. Faltings. A scalable method for multiagent constraint optimization. In *IJCAI*, pages 266–271, 2005.

[10] A. Rogers, A. Farinelli, and N. R. Jennings. Self-organising sensors for wide area surveillance using the max-sum algorithm. In *SOAR*, pages 84–100, 2009.

[11] 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.

[12] E. Rollon and J. Larrosa. Improved bounded max-sum for distributed constraint optimization. In *CP*, page (in press), 2012.

[13] R. Stranders, A. Farinelli, A. Rogers, and N. R. Jennings. Decentralised coordination of continuously valued control parameters using the max-sum algorithm. In *AAMAS*, pages 601–608, 2009.

[14] R. Stranders, A. Farinelli, A. Rogers, and N. R. Jennings. Decentralised coordination of mobile sensors using the max-sum algorithm. In *IJCAI*, pages 299–304, 2009.

[15] 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.

[16] 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.