Quality Guarantees on Locally Optimal Solutions for Distributed Constraint Optimization Problems

Jonathan P. Pearce and Milind Tambe

Computer Science Department
University of Southern California
Los Angeles, CA 90089
{jppearce, tambe}@usc.edu

Abstract. A distributed constraint optimization problem (DCOP) is a formalism that captures the rewards and costs of local interactions within a team of agents, each of whom is choosing an individual action. Because complete algorithms to solve DCOPs are not suitable for some dynamic or anytime domains, researchers have explored incomplete DCOP algorithms that result in a locally optimal solution. One metric for categorization of such algorithms, and the solutions they produce, is k-optimality; a k-optimal DCOP solution is one that cannot be improved by any deviation by k or fewer agents. This paper presents the first known guarantees on solution quality for k-optimal solutions. These guarantees are independent of the actual costs and rewards in the DCOP, and once computed can be used for any DCOP of a given graph (or hypergraph) structure in order to determine properties of the worst-case k-optimal solution for a given value of k. We provide both a mathematical proof of the soundness and experimental analysis of the tightness of the guarantees.

1 Introduction

In a large class of multi-agent scenarios, a set of agents chooses a joint action as a combination of individual actions. Often, the locality of agents' interactions means that the utility generated by each agent's action depends only on the actions of a subset of the other agents. In this case, the outcomes of possible joint actions can be compactly represented in cooperative domains by a distributed constraint optimization problem (DCOP)[1, 2]. A DCOP can take the form of a graph (or hypergraph) in which each node is an agent and each edge (or hyperedge) denotes a subset of agents whose actions, when taken together, incur costs or rewards to the agent team. Applications of DCOP include multi-agent plan coordination [3], sensor networks [1], and RoboCup soccer [4].

As the scale of these domains become large, current complete algorithms can incur large computation or communication costs. For example, a large-scale network of personal assistant agents might require global optimization over hundreds of agents and thousands of variables, which is currently very expensive. Alternatively, some domains may require that a solution be reached quickly; for example, a team of patrol units may need to quickly decide on a route for a joint patrol to efficiently survey an area before conditions change on the ground [5]. Though heuristics that significantly speed up convergence have been developed [6], the complexity is still prohibitive in large-scale domains. On the other hand, algorithms in which individual agents, or small groups of agents, react on the basis of local knowledge of neighbors and constraint utilities, lead to a system that scales up easily and is more robust to dynamic environments. Researchers have introduced *k*-optimal algorithms in which small groups of agents optimize based on their local constraints, resulting in a *k*-optimal DCOP assignment, in which no subset of *k* or fewer agents can improve the overall solution. Some examples include the 1-optimal algorithms DBA[7] and DSA[8] for Distributed CSPs, which were later extended to DCOPs with weighted constraints [2]. In [9], 2-optimal algorithms were introduced, in which optimization was done by pairs of agents coordinating to optimize based on their local constraints. While detailed experimental analyses of these algorithms on DCOPs is available[2], we still lack theoretical tools that allow us to understand the performance of such algorithms on arbitrary DCOPs.

While previous work has focused on upper bounds on the number of *k*-optima that can exist in DCOPs [5], in this paper, we provide the first known quality guarantees on *k*-optima. These guarantees are independent of the actual costs and rewards in the DCOP, and are especially useful in domains in which the costs and rewards are unknown or subject to repeated change. Because of this independence, the guarantees for *k*-optima are expressed as the proportion of all possible assignments to the DCOP that must be of equal or lesser quality to any *k*-optimal assignment.

Such guarantees could be used to help determine an appropriate *k*-optimal algorithm for agents to use, in situations where the cost of coordination between multiple agents must be weighed against the quality of the solution reached. For example, consider a team of autonomous underwater vehicles (AUVs) [10, 11] that must choose a joint action. This coordination may need to be done quickly in order to observe some transitory underwater phenomenon. The combination of individual actions by nearby AUVs may generate costs or rewards to the team, and the overall utility of the joint action is determined by their sum. Therefore, if this problem were represented as a DCOP, nearby AUVs would share constraints in the DCOP graph, while far-away AUVs would not. However, the actual costs and rewards on these constraints may not be known until shortly before the agents must act, and so, due to time constraints, an incomplete, *k*-optimal algorithm, rather than a complete algorithm, must be used to find a solution. In this case, it may be useful to know reward-independent, worst-case quality guarantees on *k*-optimal solutions for a given *k*, in order to know which algorithm to use. If increasing *k* will provide a large increase in guaranteed solution quality, it may be worth the increased computation or communication required to reach a higher *k*-optimal solution. This paper provides such guarantees for arbitrary graphs, as well as tighter guarantees that can be obtained if the structure of the DCOP graph is known ahead of time.

2 DCOP and *k*-optima

We consider a DCOP in which each agent controls a variable to which it must assign a value. These values correspond to individual actions that can be taken by the agents. Subgroups of agents, whose combined actions generate a cost or reward to the team, define the constraints between agents. Because we assume that each agent controls a single variable, we will use the terms “agent” and “variable” interchangeably, however, the results in this paper are also valid for cases in which agents control more than one variable.

Formally, we have a set of agents $\mathcal{I} := \{1, \ldots, I\}$ and a set of domains $\mathcal{A} := \{\mathcal{A}_1, \ldots, \mathcal{A}_i\}$, where the i^{th} agent takes action $a_i \in \mathcal{A}_i$. We denote the assignment of a subgroup of agents $S \subset \mathcal{I}$ by $a_S := \times_{i \in S} a_i \in \mathcal{A}_S$ where $\mathcal{A}_S := \bigtimes_{i \in S} \mathcal{A}_i$ and the assignment of the entire multi-agent team by $a = [a_1 \cdots a_I] \in \mathcal{A}$ where $\mathcal{A} := \bigtimes_{i \in \mathcal{I}} \mathcal{A}_i$. The team reward for a particular complete assignment, a, is an aggregation of the rewards obtained by the assignments to all subgroups in the team:

$$R(a) = \sum_{S \in \theta} R_S(a) = \sum_{S \in \theta} R_S(a_S)$$

where S is a minimal subgroup that generates a reward (or incurs a cost) in an n-ary DCOP (i.e. a constraint), θ is the collection of all such minimal subgroups for a given problem and $R_S(\cdot)$ denotes a function that maps $\mathcal{A}_S$ to $\mathbb{R}$. By minimality, we mean that the reward component R_S cannot be decomposed further. Mathematically:

$$\forall S \in \theta, R_S(a_S) \neq R_{S_1}(a_{S_1}) + R_{S_2}(a_{S_2})$$

for any

$$R_{S_1}(\cdot) : \mathcal{A}_{S_1} \to \mathbb{R}, R_{S_2}(\cdot) : \mathcal{A}_{S_2} \to \mathbb{R}, S_1, S_2 \subset \mathcal{I}$$

such that

$$S_1 \cup S_2 = S, S_1, S_2 \neq \emptyset.$$

It is important to express the constraints minimally to accurately represent dependencies among agents.

In [5], the *deviating group* between two assignments, a and $\tilde{a}$, was defined as

$$D(a, \tilde{a}) := \{i \in \mathcal{I} : a_i \neq \tilde{a}_i\},$$

i.e. the set of agents whose actions in $\tilde{a}$ differ from their actions in a. The *distance* between two assignments was defined as $d(a, \tilde{a}) := |D(a, \tilde{a})|$ where $| \cdot |$ denotes the cardinality of the set. An assignment a was then classified as a *k-optimum* if

$$R(a) - R(\tilde{a}) > 0 \ \forall \tilde{a} \text{ such that } d(a, \tilde{a}) \leq k.$$

In this paper, we refer to the above definition as a *strict* k-optimum, and relax the definition of a k-optimum to mean an assignment a for which

$$R(a) - R(\tilde{a}) \geq 0 \ \forall \tilde{a} \text{ such that } d(a, \tilde{a}) \leq k.$$

Equivalently, if the set of agents have reached a k-optimum, then no subgroup of cardinality $\leq k$ can improve the overall reward by choosing different actions; every such subgroup is acting optimally with respect to its context.

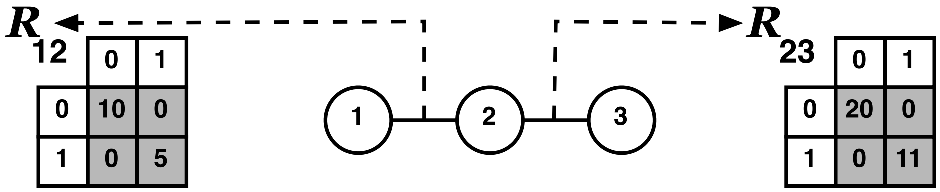

Fig. 1. DCOP example

Example 1. Figure 1 is a binary DCOP in which agents choose actions from $\{0, 1\}$, with rewards shown for the two constraints (minimal subgroups) $S_1 = \{1, 2\}$ and $S_2 = \{2, 3\}$. The assignment $a = [1\ 1\ 1]$ is 1-optimal because any single agent that deviates reduces the team reward. However, $[1\ 1\ 1]$ is not 2-optimal because if the group $\{2, 3\}$ deviated, making the assignment $\tilde{a} = [1\ 0\ 0]$, team reward would increase from 16 to 20. The globally optimal solution, $a^* = [0\ 0\ 0]$ is k-optimal for all $k \in \{1, 2, 3\}$.□

In addition to categorizing local optima in a DCOP, k-optimality provides a natural classification for DCOP algorithms. Many known algorithms are guaranteed to converge to k-optima for some $k > 0$, including DBA [2], DSA [8], and coordinate ascent [4] for $k = 1$, MGM-2 and SCA-2 [9] for $k = 2$. In these algorithms, agents or pairs of agents repeatedly choose optimal values with respect to the values chosen by neighboring agents until an equilibrium state is reached. Globally optimal DCOP algorithms such as Adopt [1], OptAPO [12] and DPOP [13] converge to a k-optimum for $k = I$.

3 Quality guarantees on *k*-optima

In this section, we provide reward-independent guarantees on solution quality for any k-optimal DCOP assignment. If we must choose a k-optimal algorithm for agents to use to quickly find an assignment to a DCOP, it is useful to see how much solution quality will be gained or lost in the worst case by choosing a higher or lower value for k. We assume that the actual numerical costs and rewards on the DCOP are not known *a priori* (if they were, then the DCOP could be solved centrally ahead of time). Since it is not possible to guarantee an actual figure for solution quality if the costs and rewards on the DCOP graph are not known, we instead provide relative guarantees. These guarantees are expressed as the proportion of all possible DCOP assignments that must have the same or lower solution quality than any k-optimal assignment.

By definition, a k-optimum a^* must be of the same or higher quality than all assignments within a distance of k. Therefore, the number of assignments of the same or lower quality can be given by the following expression, where q_i is the size of the domain for some agent i.

$$1 + \sum_{D \subset \mathcal{I}: 1 \le |D| \le k} \prod_{i \in D} (q_i - 1)$$

This expression captures a^* as well as every possible assignment $\tilde{a}$ created by a deviation of k or fewer agents from a^*. It thus follows that a k-optimum must be of the same or higher quality than the following proportion of all possible assignments to the DCOP:

$$\frac{1 + \sum_{D \subset \mathcal{I}:1 \leq |D| \leq k} \prod_{i \in D}(q_i - 1)}{\prod_{i \in \mathcal{I}} q_i}$$

If each agent's domain is the same size, given by q, this expression can be stated more simply as

$$\frac{\sum_{j=0}^{k} \binom{I}{j}(q-1)^j}{q^I}.$$

For example, a 1-optimal assignment to a DCOP of 3 agents, each with domain of size 2 must be of the same or higher quality than 50% of all assignments. To see this, consider the DCOP in Figure 1, but with unknown rewards on the links. Since there are three agents with a domain of two actions each, the total number of possible assignments is $2^3 = 8$. Suppose $a^* = [0\ 0\ 0]$ is a 1-optimal assignment. By definition, it must be of the same or higher quality than four out of the eight assignments: $[1\ 0\ 0], [0\ 1\ 0], [0\ 0\ 1]$, and itself.

4 Graph-based quality guarantees on *k*-optima

The above expressions depend only on the number of agents, the sizes of their domains, and the chosen value of k. However, we can utilize knowledge of the structure of the DCOP graph to obtain tighter quality guarantees for k-optima, expressed as the fraction of the set of all possible assignments for which a k-optima must be of equal or higher quality. The above expressions exploit the definition of k-optimality, that any k-optimal assignment a must be of equal or higher quality than any assignment $\tilde{a} \in \tilde{A}$, where $d(a, \tilde{a}) \leq k$; i.e. any assignment formed by a deviation of k or fewer agents. By using the structure of the DCOP graph, we can expand the set $\tilde{A}$ to include assignments that are a distance greater than k from a, thus providing a stronger guarantee on the quality of a with respect to all possible assignments to the DCOP. If a^* is a k-optimal assignment to a DCOP, then any assignment where *any number* of disjoint subsets of size $\leq k$ have deviated from their actions in a^* must be of the same or lower quality as a^*, as long as no constraint exists in the DCOP graph or hypergraph between any two agents in different such subsets. For example, consider again the DCOP in Figure 1, but with unknown rewards on the links, and suppose $a^* = [0\ 0\ 0]$ is a 1-optimal assignment. Since a deviation by any individual agent cannot increase the overall reward, and agents 1 and 3 do not share any constraints, a deviation by both these agents also cannot increase the overall reward. So we can add assignment $[1\ 0\ 1]$ to the list of assignments of lesser or equal quality to a^*, and can provide a better guarantee: any 1-optimum in a DCOP with this graph structure must be of the same or higher quality than 62.5% of all possible assignments as opposed to 50%.

To illustrate this idea formally we introduce the following notation. If we define n different subsets of agents as D_i for $i = 1 \ldots n$, we use $D^m = \cup_{i=1}^{m} D_i$, i.e. D^m is the union of the first m subsets. The following proposition allows us to enumerate the assignments with equal or lower reward than $R(a^*)$. The proof is by induction over each subset D_i for $i = 1 \dots n$.

Proposition 1. *Let a^* be a k-optimal assignment. Let a^n be another assignment for which $D(a^*, a^n)$ can be expressed as $D^n = \cup_{i=1}^{n} D_i$ where:*

- $\forall D_i, |D_i| \leq k$ (subsets contain k or fewer agents)
- $\forall D_i, D_j, \nexists i \in D_i, j \in D_j$ *such that* $i = j$ (subsets are disjoint)
- $\forall D_i, D_j, \nexists i \in D_i, j \in D_j$ *such that* $i, j \in S$, *for any* $S \in \theta$. (no constraint exists between agents in different subsets)

Then, $R(a^*) \geq R(a^n)$.

Proof.

Base case: If $n = 1$ then $D^n = D_1$ and $R(a^*) \geq R(a^n)$ by definition of k-optimality.

Inductive step: $R(a^*) \geq R(a^{n-1}) \Rightarrow R(a^*) \geq R(a^n)$.

If D^n is given according to the above conditions, then the set of all agents can be divided into the set of agents in D^{n-1}, the set of agents in D_n, and the set of agents not in D^n, and the global reward for any assignment a can be given according to the following expression:

$$R(a) = \sum_{S \in \theta : S \cap D^{n-1} \neq \emptyset} R_S(a_S) + \sum_{S \in \theta : S \cap D_n \neq \emptyset} R_S(a_S) + \sum_{S \in \theta : S \cap D^n = \emptyset} R_S(a_S).$$

By inductive hypothesis, $R(a^*) \geq R(a^{n-1})$. Therefore,

$$\sum_{S \in \theta : S \cap D^{n-1} \neq \emptyset} R_S(a^*_S) + \sum_{S \in \theta : S \cap D_n \neq \emptyset} R_S(a^*_S) + \sum_{S \in \theta : S \cap D^n = \emptyset} R_S(a^*_S)$$
$$\geq \sum_{S \in \theta : S \cap D^{n-1} \neq \emptyset} R_S(a^{n-1}_S) + \sum_{S \in \theta : S \cap D_n \neq \emptyset} R_S(a^{n-1}_S) + \sum_{S \in \theta : S \cap D^n = \emptyset} R_S(a^{n-1}_S)$$

which reduces to

$$\sum_{S \in \theta : S \cap D^{n-1} \neq \emptyset} R_S(a^*_S) \geq \sum_{S \in \theta : S \cap D^{n-1} \neq \emptyset} R_S(a^{n-1}_S)$$

because the rewards generated by agents outside D^{n-1} are the same for a^* and a^{n-1}.

Now, let $\hat{a}$ be an assignment such that $D(a^*, \hat{a}) = D_n = D(a^{n-1}, a^n)$. By the definition of k-optimality, $R(a^*) \geq R(\hat{a})$. Therefore,

$$\sum_{S \in \theta : S \cap D^{n-1} \neq \emptyset} R_S(a^*_S) + \sum_{S \in \theta : S \cap D_n \neq \emptyset} R_S(a^*_S) + \sum_{S \in \theta : S \cap D^n = \emptyset} R_S(a^*_S)$$
$$\geq \sum_{S \in \theta : S \cap D^{n-1} \neq \emptyset} R_S(\hat{a}_S) + \sum_{S \in \theta : S \cap D_n \neq \emptyset} R_S(\hat{a}_S) + \sum_{S \in \theta : S \cap D^n = \emptyset} R_S(\hat{a}_S).$$

which reduces to

$$\sum_{S \in \theta : S \cap D_n \neq \emptyset} R_S(a^*_S) \geq \sum_{S \in \theta : S \cap D_n \neq \emptyset} R_S(\hat{a}_S)$$

because the rewards generated by agents outside D_n are the same for a^* and $\hat{a}$.

We also know that

$$\sum_{S\in\theta:S\cap D^n=\emptyset} R_S(a^*_S) = \sum_{S\in\theta:S\cap D^n=\emptyset} R_S(a^n_S)$$

because the rewards generated by agents outside D^n are the same for a^* and a^n.

Therefore,

$$\begin{aligned} R(a^*) &= \sum_{S\in\theta:S\cap D^{n-1}\neq\emptyset} R_S(a^*_S) + \sum_{S\in\theta:S\cap D_n\neq\emptyset} R_S(a^*_S) + \sum_{S\in\theta:S\cap D^n=\emptyset} R_S(a^*_S) \\ &\geq \sum_{S\in\theta:S\cap D^{n-1}\neq\emptyset} R_S(a^{n-1}_S) + \sum_{S\in\theta:S\cap D_n\neq\emptyset} R_S(\hat{a}_S) + \sum_{S\in\theta:S\cap D^n=\emptyset} R_S(a^n_S) \\ &= \sum_{S\in\theta:S\cap D^{n-1}\neq\emptyset} R_S(a^n_S) + \sum_{S\in\theta:S\cap D_n\neq\emptyset} R_S(a^n_S) + \sum_{S\in\theta:S\cap D^n=\emptyset} R_S(a^n_S) \\ &= R(a^n) \end{aligned}$$

because the rewards generated by agents in D^{n-1} are the same for a^{n-1} and a^n, and the rewards generated by agents in D_n are the same for $\hat{a}$ and a^n. ■.

This proposition makes it possible to compute graph-based guarantees on solution quality for k-optima by testing every possible assignment of actions to agents for the specified conditions. While this process requires time exponential in the number of agents, it need only be performed once for any particular DCOP graph, and can then be used for any DCOP (with any costs and rewards on the constraints) that uses that graph.

5 Experimental results

We performed three sets of experiments. In the first set, we computed theoretical quality guarantees for k-optima in several common graph structures, for up to 25 agents, based on the finding in Proposition 1, to observe the different guarantees possible for different graph structures. In the second set, we began with a full graph, and systematically removed constraints until a chain graph was left, computing graph-based and non-graph-based quality guarantees at each step to more closely examine the effect of graph structure on the guarantees. Finally, in the third set of experiments, actual DCOPs were generated, and the quality of k-optima in those DCOPs were compared to the guarantees. This was done, first to verify that the guarantees hold, and second, to examine the tightness of the guarantees compared to the solution quality of experimentally generated k-optima.

The results of the first set of experiments is shown in Figures 2-5. Figure 2 shows the quality guarantees made possible by Proposition 1 for k-optima in DCOPs with fully connected graphs. Figure 2(a) is for agents with a domain of two actions, and (b) is for agents with a domain of three actions; the number of agents is plotted on the x-axis. Ten data series, showing quality guarantees for $k = \{1, 2, \ldots, 10\}$, are plotted on the y-axis; the lower-leftmost data series is for $k = 1$, and the upper-rightmost data series is for $k = 10$. So, for example, for a DCOP with 5 agents and a fully-connected graph, a 1-optimum must be of the same or higher quality than 19% of all possible

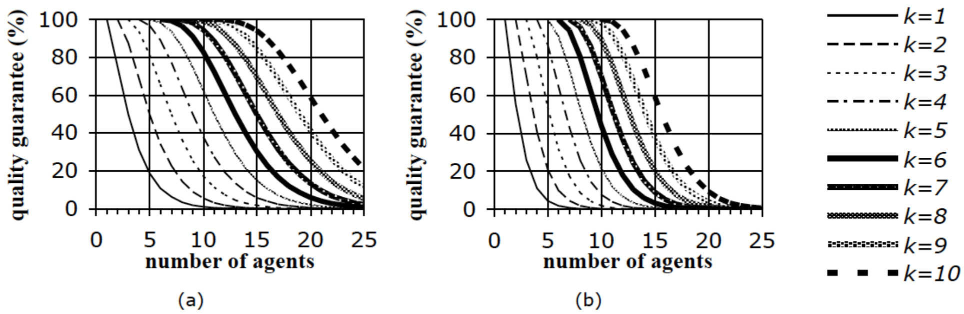

Fig. 2. Quality guarantees of k-optima for fully connected DCOP graphs of agents with domains of (a) 2 and (b) 3 actions.

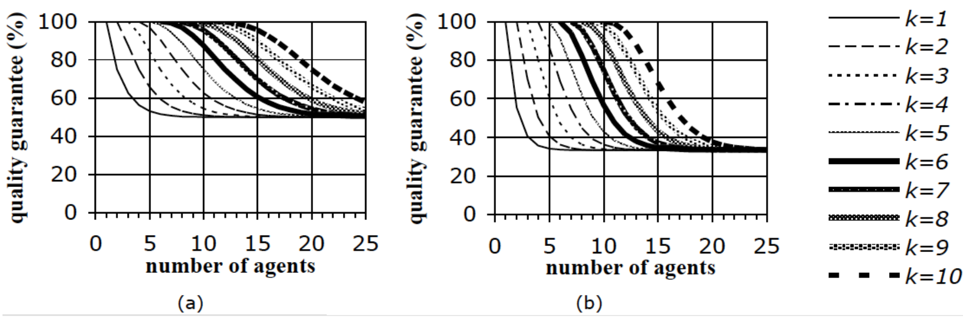

Fig. 3. Quality guarantees of *k*-optima for star DCOP graphs of agents with domains of (a) 2 and (b) 3 actions.

assignments; a 2-optimum must be the same or better than 50%; a 3-optimum: 81%;, and a 4-optimum: 97%. A 5-optimum represents the globally optimal solution, which is of the same or higher quality than 100% of all possible assignments.

Figures 3, 4, and 5 show the same information for *k*-optima in graphs that are stars (constraints exist between one agent and each other agent), chains, and binary trees. Note that the guaranteed effect of increasing the chosen value of *k* can vary dramatically from one type of graph to another - for example, moving from $k = 1$ to $k = 2$ for a star of 10 agents means moving from a guarantee of 50% to a guarantee of 51%, while for a chain, the same change in *k* means moving from a guarantee of 14% to 49%.

In the second set of experiments, we began with a fully-connected graph of 10 agents, numbered 1 to 10. Constraints were then repeatedly removed from this graph according to the following algorithm until only a chain remained: (1) Choose the most connected agent *i* from the set of all agents. (2) Choose the most connected agent *j* from the set of *i*'s neighbors, not including the node whose ID immediately follows that of *i*. At each step, non-graph-based and graph-based quality guarantees were computed, for each value of *k* from $\{1 \ldots 9\}$. Results of this experiment are shown in Figure 6. The gray line represents the quality guarantee possible for *k*-optima if the DCOP graph

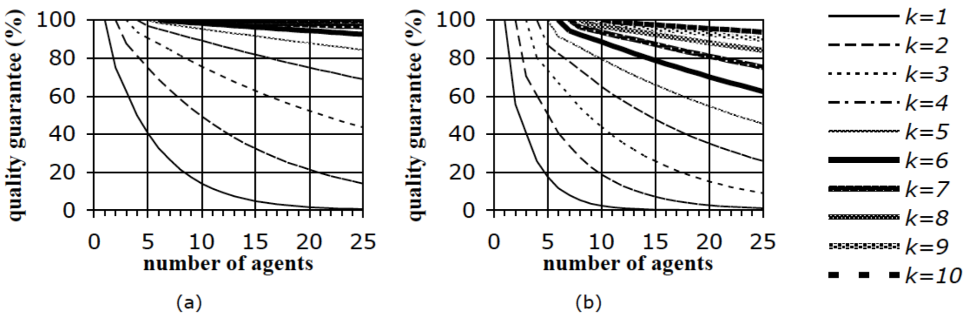

Fig. 4. Quality guarantees of *k*-optima for chain DCOP graphs of agents with domains of (a) 2 and (b) 3 actions.

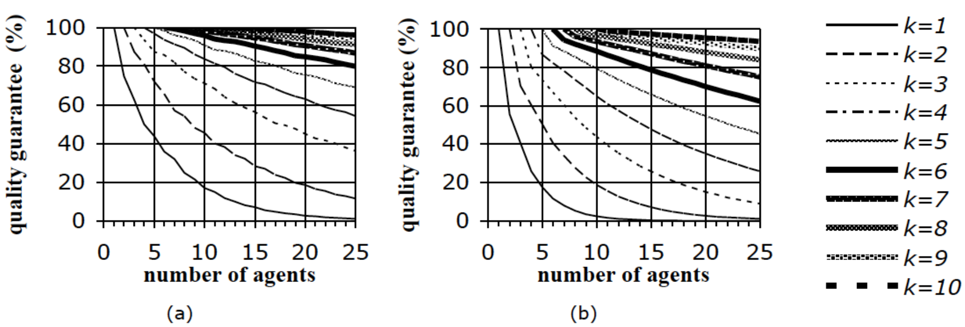

Fig. 5. Quality guarantees of *k*-optima for binary tree DCOP graphs of agents with domains of (a) 2 and (b) 3 actions.

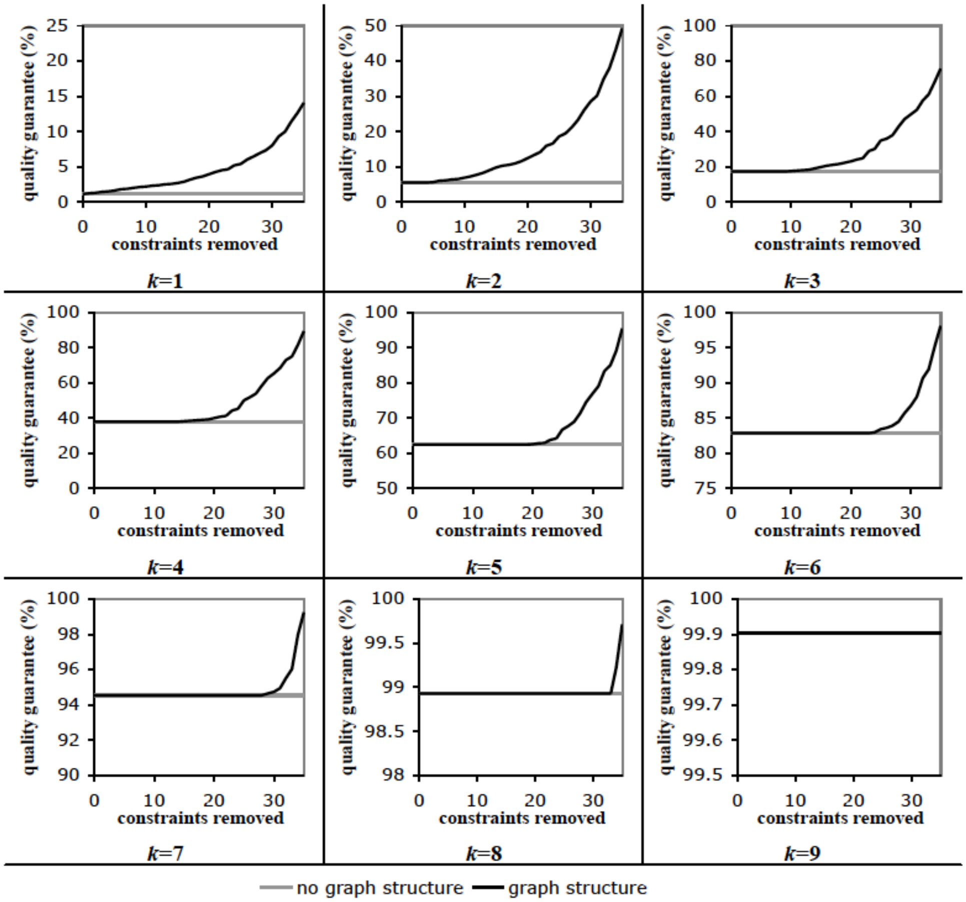

Fig. 6. Graph-based vs. non-graph-based quality guarantees for a DCOP of 10 agents.

structure is not known or not considered, and the black line represents the guarantee possible if it is considered; removing constraints from the graph only has an effect in the latter case. In general, the effect of removing constraints has very little effect at first, but after a certain point, dramatically increases the quality guarantee possible for k-optima. The effect of removing constraints was most dramatic for $k = 3$. For example, in a graph with 30 constraints removed, if the DCOP graph structure is not considered, a 3-optimum can only be guaranteed to be of the same or higher quality than 17% of all possible DCOP assignments. However, considering the graph structure reveals that any 3-optimum must actually be of the same or higher quality than 39% of all possible assignments.

While the graph-based guarantees were proven to be sound in the previous section, they represent a worst case over all possible costs and rewards that can exist on the constraints in a given graph. The third set of experiments compares the guarantees with the solution quality of actual k-optima in graphs with randomly generated costs and rewards on the constraints. Four common types of DCOP graphs were generated (fully connected, chains, stars, and binary trees) for systems of three to six agents, each with a domain of two actions. Then, 100,000 DCOPs with random rewards were generated for each graph. This was done by generating a reward from a uniform random distribution from $[0 \ldots 1]$, for each constraint function in the DCOP. In each of the 100,000 runs, all k-optima for $k = \{1 \ldots I\}$ were found, and their rankings in solution quality over all assignments in that DCOP were computed. The k-optima with the lowest rankings over all 100,000 runs for each k are plotted for each graph examined in Figure 7.

In each graph, k is plotted on the x-axis. The quality guarantee, and the actual quality of the worst-ranked k-optimum are plotted on the y-axis. For example, in a chain of six agents, the quality guarantee says that any 2-optima must be of the same or higher quality than 69% of all possible solutions in its DCOP, but the worst 2-optimum actually generated was of the same or higher quality than 78% of all solutions in its DCOP. In most cases, the guarantees exactly match the experimental values (they do match for any graph of 3 agents, and for any star-shaped graph); the cases where they do not could be explained by the decreasing likelihood of randomly generating constraints with rewards that ensure a worst-case quality k-optimum as the number of constraints in the problem increases. Most importantly, these experimental results suggest that the quality guarantees made possible by Proposition 1 are provably tight; this proof is clearly an area for future work.

6 Related work and conclusion

In [5], upper bounds on the number of possible k-optima that could exist in a given DCOP graph were presented. The work in this paper focuses instead on lower bounds on solution quality for k-optima for a given DCOP graph. This paper provides a complement to the experimental analysis of local minima (1-optima) arising from the execution of incomplete DCOP algorithms [2, 9]. However, in this paper, the emphasis is on the worst case rather than the average case.

We note that k-optimality and the guarantees presented in this paper can also apply to centralized constraint reasoning. However, examining properties of solutions that arise from coordinated value changes of small groups of variables is especially useful in distributed settings, given the computational and communication expense of large-scale coordination.

The guarantees presented in this paper are the first known guarantees on solution quality for k-optimal solutions, and are independent of the actual costs and rewards in the DCOP. Once computed, they can be used for any DCOP of a given graph (or hypergraph) structure in order to determine properties of the worst-case k-optimal solution for a given value of k. Both a mathematical proof of the soundness and an experimental analysis of the tightness of the guarantees was provided.

References

1. Modi, P.J., Shen, W., Tambe, M., Yokoo, M.: Adopt: Asynchronous distributed constraint optimization with quality guarantees. Artificial Intelligence **161**(1-2) (2005) 149–180

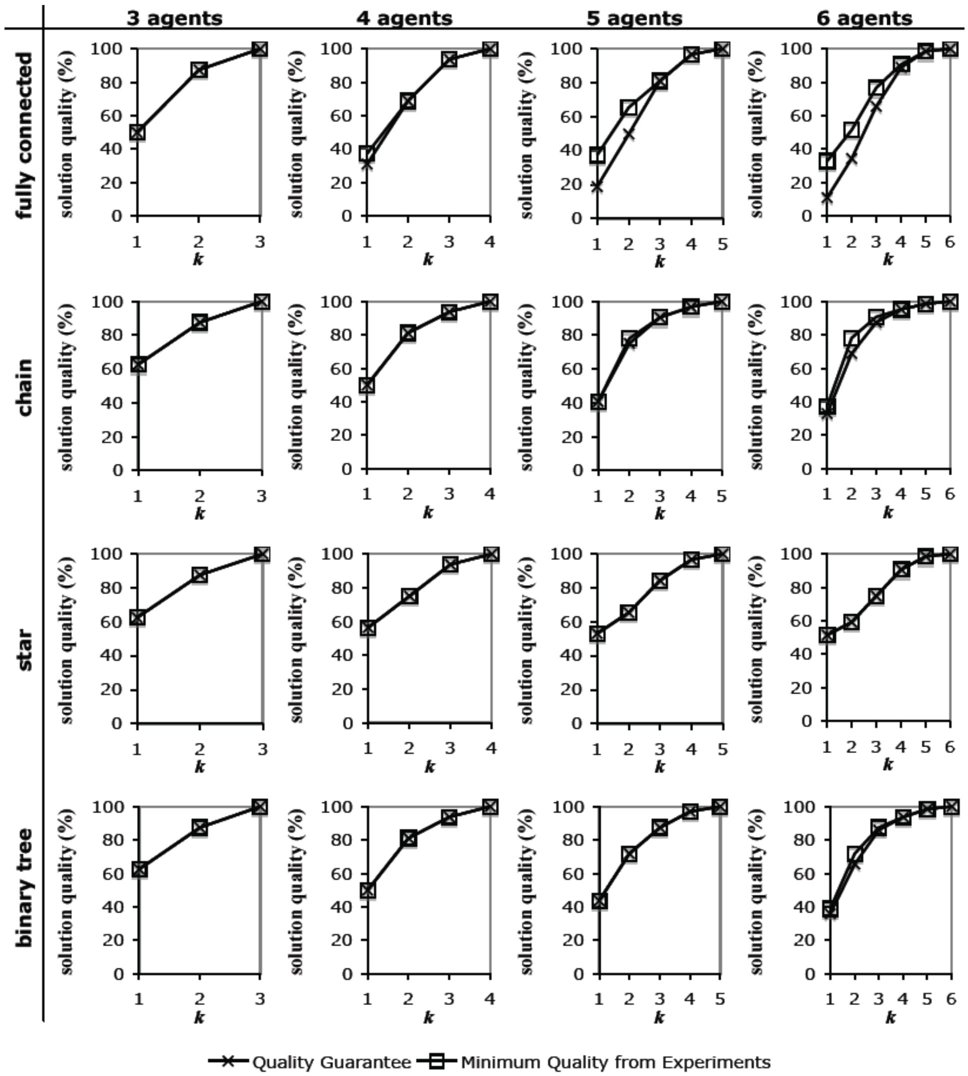

Fig. 7. Comparison of theoretical guarantees with actual minimum quality *k*-optima from experiments

2. Zhang, W., Xing, Z., Wang, G., Wittenburg, L.: An analysis and application of distributed constraint satisfaction and optimization algorithms in sensor networks. In: AAMAS. (2003)
3. Cox, J., Durfee, E., Bartold, T.: A distributed framework for solving the multiagent plan coordination problem. In: AAMAS. (2005)
4. Vlassis, N., Elhorst, R., Kok, J.R.: Anytime algorithms for multiagent decision making using coordination graphs. In: Proc. Intl. Conf. on Systems, Man and Cybernetics. (2004)
5. Pearce, J.P., Maheswaran, R.T., Tambe, M.: Solution sets for DCOPs and graphical games. In: AAMAS. (2006)
6. Maheswaran, R.T., Tambe, M., Bowring, E., Pearce, J.P., Varakantham, P.: Taking DCOP to the real world: efficient complete solutions for distributed multi-event scheduling. In: AAMAS. (2004)
7. Yokoo, M., Hirayama, K.: Distributed breakout algorithm for solving distributed constraint satisfaction and optimization problems. In: ICMAS. (1996)
8. Fitzpatrick, S., Meertens, L.: Distributed coordination through anarchic optimization. In Lesser, V., Ortiz, C.L., Tambe, M., eds.: Distributed Sensor Networks: A Multiagent Perspective. Kluwer (2003) 257–295
9. Maheswaran, R.T., Pearce, J.P., Tambe, M.: Distributed algorithms for DCOP: A graphical-game-based approach. In: PDCS. (2004)
10. Zhang, Y., Bellingham, J.G., Davis, R.E., Chao, Y.: Optimizing autonomous underwater vehicles' survey for reconstruction of an ocean field that varies in space and time. In: American Geophysical Union, Fall meeting. (2005)
11. Rajan, K.: private communication. (2005)
12. Mailler, R., Lesser, V.: Solving distributed constraint optimization problems using cooperative mediation. In: AAMAS. (2004)
13. Petcu, A., Faltings, B.: A scalable method for multiagent constraint optimization. In: IJCAI. (2005)