PKOPT: Faster k-Optimal Solution for DCOP by Improving Group Selection Strategy

Elnaz Bigdeli, Maryam Rahmaninia
Mathematic and Computer Science Department
Institute for Advanced Studies in Basic Sciences,
Zanjan, Iran
Email:{e_bigdeli,m_rahmani}@iasbs.ac.ir

Mohsen Afsharchi
Department of Electrical and Computer Engineering
University of Zanjan
Zanjan, Iran
Email: afsharchim@iasbs.ac.ir

Abstract*—A significant body of work in multiagent systems over more than two decades has focused on multi-agent coordination (1). Many challenges in multi-agent coordination can be modeled as Distributed Constraint Optimizations (DCOPs). Many complete and incomplete algorithms have been introduced for DCOPs but complete algorithms are often impractical for large-scale and dynamic environments which lead to study incomplete algorithms. Some incomplete algorithms produce k-optimal solutions; a k-optimal solution is the one that cannot be improved by any deviation by k or fewer agents. In this paper we focus on the only k-optimal algorithm which works for arbitrary k, entitled as KOPT. In both complete and incomplete algorithms, computational complexity is the major concern. Different approaches are introduced to solve this problem and improve existing algorithms. The main contribution of this paper is to decrease computational complexity of KOPT algorithm by introducing a new method for selecting leaders which should assign new values to a group of agents. This new approach is called Partial KOPT (PKOPT). PKOPT is an effective method to reduce computational load and power consumption in implementation. This paper under various assumptions presents an analysis of sequential and stochastic PKOPT algorithms.

***Keywords*-Multi Agent Systems; Distributed Constraint Optimization (DCOP); k-optimality**

I. Introduction

Multi-agent systems are a popular way to model complex interactions and coordination required to solve distributed problems. A multi-agent system is a network of cooperative agents used to perform distributed computation. Networks of cooperative agents are heterogeneous and not all agents have direct communication link to one another. Additionally, information is distributed throughout the network either due to privacy concerns or impractically of centralizing. In this network each agent is autonomous entity with local information and has ability to perform an action in cooperative situations in which agents collaborate to achieve a common goal.

Agents need to coordinate their activities to accomplish their collective goals. Distributed Constraint Optimization (DCOP) is a common formalism to represent multi-agent systems in which agents cooperate to optimize a global objective (2), (3). Distributed Constraint Optimization (DCOP) has been applied to different domains. DCOPs are able to model the task of scheduling meetings in large organizations, where privacy needs make centralized constraint optimization difficult (4). DCOPs are also able to model the task of allocating sensor nodes to targets in sensor networks, where the limited communication and computation power of individual sensor nodes makes centralized constraint optimization difficult (5). Finally, DCOPs are able to model the task of coordinating teams of unmanned vehicles in disaster response scenarios, where the need for rapid local responses makes centralized constraint optimization difficult (6).

There are two main categories for DCOP algorithms, Complete and incomplete algorithms. Complete algorithms, are algorithms that always find a configuration of variables that maximizes the global objective function. Adopt (Asynchronous Distributed OPTimization) (5) and DPOP (Dynamic Programming OPtimisation) (3) are two well known complete algorithms. In contrast, incomplete algorithms find semi optimal solutions and do not guarantee to achieve global optimal solution. Algorithms such as Max-Sum (7), Distributed Arc Consistency (8) and KOPT (9) are in this category.

k-optimal algorithms guarantee to provide solutions that cannot be improved by any group of k or fewer agents changing their decision. Many incomplete algorithms such as MGM1 (10) and DSA1 (11) yield 1-optimal solutions. The other version of MGM1 is MGM2 (12) which provides 2-optimal solutions. KOPT algorithm (9) is the only incomplete algorithm which works for arbitrary k. The main focus of this paper is on KOPT algorithm. In spite of many advantages of KOPT algorithm, it suffers from a major problem like other DCOP algorithms. In some environments with high degree of interactions, computational complexity of KOPT algorithm is not tolerable. In this paper we propose an algorithm that decreases the computational complexity in substantial amount.

The structure of the paper is as follows: In section II, formal definitions of DCOP and k-optimal solutions are presented. In section III, KOPT algorithm and its main issues are described. Sequential and stochastic PKOPT algorithms and their main issues are discussed in section IV. In section V, computational complexity of sequential and stochastic PKOPT algorithms are analyzed. Finally, experimental results of new algorithms and their comparison with KOPT algorithm are depicted in section VI.

II. Background

In this section, we will provide some basic definitions about DCOP and k-Optimality.

A. Distributed Constraint Optimization (DCOP)

A DCOP is defined by a set of variables $\mathcal{V} = \{v_1, \dots, v_n\}$, a set of discrete finite domains for each variable $\mathcal{D} = \{D_1, \dots, D_n\}$ and a set of constraints $\mathcal{C} = \{c_1, \dots, c_q\}$. Each variable is controlled by a separate agent that can communicate with other agents. A joint assignment $\mathcal{A} = \{a_1, \dots, a_n\}$ specifies a value for each variable, in which a_i is the value of agent i. Each constraint includes a set of variables and based on the values which each agent chooses, a constraint defines a real-valued cost. In this paper only binary constraints are considered. It means that each constraint has two variables. Thus, for each pair of variables v_i, v_j, we will be given a cost function $\mathcal{F}_{ij} : \mathcal{D}_i \times \mathcal{D}_j \longrightarrow \Re$ which determines the value of a constraint. If there is no constraint between v_i, v_j, function $\mathcal{F}$ will be 0. A cost function takes values of variables as input and returns a value as a non-negative number for a constraint. Utility of agent i for assignment $\mathcal{A}$ is:

$$\mathcal{U}_i(\mathcal{A}) = \sum_{v_j \in \mathcal{V}} \mathcal{F}_{ij}(a_i, a_j)$$
$$\text{Where } v_i \leftarrow a_i, v_j \leftarrow a_j, a_i, a_j \in \mathcal{A} \tag{1}$$

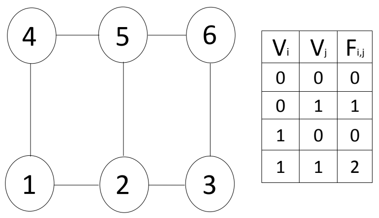
It means that the utility of i_{th} agent is the sum of the cost functions of all the constraints which an agent belongs to.

The goal is to choose values for variables such that a given objective function is maximized. The objective function is described as the sum over a set of cost functions, or valued constraints. Thus the objective is to maximize:

$$\mathcal{R}(\mathcal{A}) = \sum_{(v_i, v_j) \in \mathcal{V}} \mathcal{F}_{ij}(a_i, a_j)$$
$$\text{Where } v_i \leftarrow a_i, v_j \leftarrow a_j, a_i, a_j \in \mathcal{A} \tag{2}$$

$\mathcal{R}(\mathcal{A})$ is a solution quality for a joint assignment $\mathcal{A}$ (13) (5).

Figure 1 shows an example DCOP with 6 variables and 7 constraints with identical cost function.



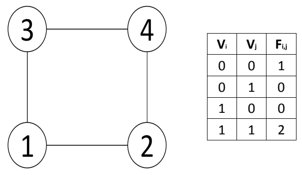
V_i	V_j	$F_{i,j}$
0	0	0
0	1	1
1	0	0
1	1	2

Figure 1. An example DCOP with six binary variables. Each constraint has the same cost function.

B. k-Optimality

Let $\mathcal{D}(\mathcal{A}, \mathcal{A}')$ denotes a set of variables with different values in $\mathcal{A}$ and $\mathcal{A}'$. A DCOP assignment $\mathcal{A}$ is k-optimal if $\mathcal{R}(\mathcal{A}) \geq \mathcal{R}(\mathcal{A}')$ for all $\mathcal{A}'$ for which $\mid \mathcal{D}(\mathcal{A}, \mathcal{A}') \mid \leq k$. Where $\mid \mathcal{D} \mid$ denotes the cardinality of set $\mathcal{D}$ (13).

Example: Consider the graph in Figure 2 in which $\mathcal{V} = \{v_1, v_2, v_3, v_4\}$ are variables with binary domain. In this example constrains are $c_1 = \{v_1, v_2\}, c_2 = \{v_1, v_3\}, c_3 = \{v_2, v_4\}, c_4 = \{v_3, v_4\}$. $\mathcal{A}_1 = \{0, 0, 0, 0\}$ is a 1-optimal assignment with $\mathcal{R}(\mathcal{A}_1) = 4$. This assignment is 1-optimal because no unitary change in assignment $\mathcal{A}_1$ improves the reward. But it is not 2-optimal assignment because if one pair of agents change their values simultaneously to 1 the solution quality will increase. Assignment $\mathcal{A}_2 = \{1, 1, 1, 1\}$ with $\mathcal{R}(\mathcal{A}_2) = 8$ is 4-optimal assignment and also is the best assignment.



V_i	V_j	$F_{i,j}$
0	0	1
0	1	0
1	0	0
1	1	2

Figure 2. An example DCOP with four binary variables. Each constraint has the same cost function.

There is a detailed investigation in the field of k-optimal algorithms. Pearce and Tambe presented the first known guarantees on solution quality for k-optimal solutions (14). They provided reward and structure-independent guarantee on solution quality for any k-optimal DCOP assignment. In addition, tighter guarantee for ring and star graphs were presented in their work.

Two main properties of k-optimal solutions are introduced in (15). The first one is the worst-case guarantee on the solution quality of k-optima in a DCOP. The second one is the worst-case guarantee on the number of k-optimal solutions that can exist in a DCOP. A close view to the k-optimal solution set can be found in (15), in which using coding theory they provide an upper bound for k-optimal solution set.

III. KOPT ALGORITHM

KOPT algorithm as an incomplete algorithm is introduced by Katagishi which is the first DCOP incomplete algorithm for arbitrary k (9). This algorithm consists of 3 phases. In phase 1, every agent gathers information from its neighbors. In phase 2, every agent form a group randomly and calculates the best value assignment to its group members by using the information acquired in phase 1. Then every agent broadcasts the assignment to all its neighbors. In phase 3, every agent selects the best value assignment which has the highest utility among the assignments sent by its nearby agents. If all the agents in the assignment $\mathcal{A}$ know that all the agents in $\mathcal{A}$ have chosen $\mathcal{A}$, they will change their variables according to $\mathcal{A}$. Otherwise, none of the agents in $\mathcal{A}$ will move. These three phases make up one round.

The most important point of this algorithm is to choose group members. How to choose group members is important to guarantee the k-optimality of the solution. Each group has a leader which locates at the center of the group. Each group includes k active agents and some static agents. Active agents are the agents which can change their value to achieve the highest utility in their groups. Static agents are located at boundary of groups and they cannot change their values to ensure that the global utility is always strictly increasing in KOPT.

Based on KOPT algorithm each agent starts from its neighborhood and extends its group until it includes k active agents. Each agent can form different groups, but according to KOPT algorithm groups are selected randomly.

For the graph in Figure 1 and $k = 2$, groups will be as follow. Leaders of groups are shown in bold and active agents are in italic.

$\text{Group}_1 = \{\mathbf{1}, \mathit{2}, 3, 4, 5\}$
$\text{Group}_2 = \{1, \mathbf{2}, \mathit{3}, 5, 6\}$
$\text{Group}_3 = \{2, \mathbf{3}, 5, \mathit{6}\}$
$\text{Group}_4 = \{\mathit{1}, 2, \mathbf{4}, 5\}$
$\text{Group}_5 = \{1, 2, \mathit{4}, \mathbf{5}, 6\}$
$\text{Group}_6 = \{2, 3, 4, \mathit{5}, \mathbf{6}\}$

A. KOPT issues

Although KOPT is an effective algorithm to solve DCOP problems, it suffers form some drawbacks. The first one is that, for $k < n$ (n is the number of agents) this algorithm does not guarantee to obtain global optimal solution and finding global optimal solution by this approach is possible by increasing k which in return increases computational complexity.

The second drawback is that in KOPT algorithm groups are formed randomly for each agent. Hence there is no pre specified method to choose the best group to reach the best possible solution (9). Group formation is important to guarantee the k-optimality of solution. Different group formations can lead to different solutions. In other words, groups in DCOP can be in a form which leads to solutions with low quality.

The third, and in our opinion, the most important drawback is that all leaders start to calculate the best possible assignment for their group members but, in the end of each round, to avoid conflicts among leader assignments, some of these leaders are chosen and their assignments are set. It means that each agent belongs to different groups and it should finally choose one of the assignment of the leaders. Based on KOPT algorithm the best possible assignment is chosen for each agent and other assignments are ignored. As it is clear, finding the best assignment in each group needs a large number of messages to be sent and received which increases computational complexity.

Despite the recent improvements in KOPT algorithm, solving DCOP in a fully connected network with large number of agents is still a very challenging task. Specially, in graphs such as mesh or complete structures, where network has large number of agents and communications among agents are hundred of thousands, KOPT algorithm cannot converge after finite number of rounds. In this paper, we present a generalization of KOPT algorithm to overcome the third problem, by using two partial updating approaches. Partial updating is an effective method to reduce the computational complexity while keep the solution quality almost the same.

By using partial approach, complexity can be decreased by dismissing leaders which their assignments will be ignored in the end of each round. Consequently, there will be a substantial decrease in computational complexity. However, this approach reduces the computational complexity but may increase the time of convergence.

IV. PARTIAL KOPT (PKOPT) ALGORITHMS

Based on our discussion from section III, in phase 2 of KOPT algorithm all the leaders calculate a new assignment for their group members whereas some of these assignments will be ignored to avoid conflicts among leader assignments. Hence, it is better to have some leaders not to compute new assignments. To this end, permission to calculate new assignment is granted to some leaders in each round. The leaders which receive permission to compute new assignments are called active leaders. It is worthwhile to mention that active leaders are different from active agents. Active agents are those that can change their value to the value which leaders send to them and they do not perform any computation. As it is described in KOPT algorithm, there are n different groups for a graph with n agents. Consequently, there are n leaders. We define the index set $\mathcal{L} = \{1, 2, .., n\}$. The index set $\mathcal{L}$ is divided into h subsets $\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_h\}$. Each subset includes agent's ID which should be activated. In each round based on a pre-defined approach which we discuss it in next section, one of the $\mathcal{S}_i$ is selected and the leaders in $\mathcal{S}_i$ run a complete algorithm in a synchronous manner. The main problem is to assign agents to subgroups $\{\mathcal{S}_1, \dots, \mathcal{S}_h\}$. The best approach to this end is the one which assign leaders in subsets which by activation of theirs leaders solution quality increases in each round. Such an approach is the most desired one but in return increases the complexity of algorithm. Leaders' selection can be performed by a sequential or stochastic approach. These two approaches are simple methods which decrease computational complexity in substantial amount.

In sequential approach, in the first round, $\mathcal{S}_1$ is selected, in the second round, $\mathcal{S}_2$ and in the h_{th} round S_h are selected, in round $h+1$, $\mathcal{S}_1$ is selected again. Consequently, after h rounds all subsets are selected exactly once.

In stochastic approach with a probability $\mathcal{P}$, a subset $\mathcal{S}_i$ is selected. With respect to the law of large numbers, given a large number of rounds all the subsets will be selected with equal probability (16). Therefore, each subset will be selected finally after a large number of rounds and all the leaders will have the chance to be activated. In the following, the proposed approaches are described in more detail.

A. Sequential PKOPT

In this approach at a given round r, one of the subsets $\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_h$ is chosen in a sequential fashion. After h rounds each subset is selected only once. In this method, we assume in each round, each leader in a network, synchronously checks if

$$\ell \ \% \ h = r \ \% \ h \tag{3}$$

Where ℓ is leader's ID and r is the number of round. Then this leader begins its computation based on KOPT algorithm for the best assignment.

As an example consider the graph in Figure 1. For $h = 3$, set $\mathcal{S}$ will be:

$$\mathcal{S} = \{\{\mathcal{A}g_1, \mathcal{A}g_4\}, \{\mathcal{A}g_2, \mathcal{A}g_5\}, \{\mathcal{A}g_3, \mathcal{A}g_6\}\}$$

Based on the above mentioned approach, in round 1, $\{\mathcal{A}g_1, \mathcal{A}g_4\}$, in round 2 $\{\mathcal{A}g_2, \mathcal{A}g_5\}$ and in round 3 $\{\mathcal{A}g_3, \mathcal{A}g_6\}$ are selected and it continues in the same fashion. It is obvious that in the end of h_{th} round all subsets are selected exactly once. Obviously different methods can be used in the sequential approach.

B. Stochastic PKOPT

Stochastic approach is similar to the sequential approach. The only difference is that at a given round r one of the sets $\mathcal{S}_i$ is sampled randomly from $\{\mathcal{S}_1, \dots, \mathcal{S}_{2^n-1}\}$ with probability $\frac{1}{2^n-1}$. In the stochastic approach each leader generates a random number from [0,1] and if this number is more than a threshold θ this leader is activated. All the leaders in a graph act synchronously and in a moment there will be leaders which are activated. Then, each active leader compute the best assignment for its group members based on KOPT algorithm.

V. Computational Complexity Analysis

In sequential algorithm based on description in IV-A in each round a subset which consists of leaders' id is selected. Number of subsets h is an important parameter in this algorithm and influences computational complexity, solution quality and number of rounds. To see how h influences these parameters we set up an experiment for graphs with different structures and $n = 22$ to show computational complexity is decreased by increasing h in each round, by applying our PKOPT algorithm. The results are shown in Figure 3. In set $\mathcal{S} = \{\mathcal{S}_1, \dots, \mathcal{S}_h\}$ increasing h means increasing cardinality of $\mathcal{S}$. In a graph with n agents there are $\lceil \frac{n}{h} \rceil$ leaders in each subset. Hence, by increasing h, number of leaders in each subset will be decreased. As it is described, in each round a subset is selected, and if the leaders of a subset decreases, the number of leaders which will be activated in each round will be few. Therefore number of leaders which compute new assignments are few and as a result computational complexity will be decreased.

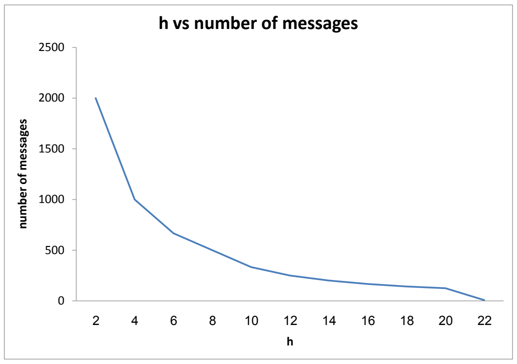


Figure 3. Total number of messages are decreased by increasing h

As it is depicted in Figure 4 by increasing h solution quality will be decreased. By increasing h number of leaders which are activated will be decreased. Hence, the number of new assignment which are computed will be decreased and to reach the specified quality more number of rounds will be needed. It means that, to reach the equal solution quality in the same number of rounds like KOPT algorithm, h should not be increased much.

Figure 5 depicts the relation between h and number of rounds, it shows that by increasing h, the number of rounds will be increased. As it is mentioned before, by increasing h, the number of leaders which should be activated in each round will be decreased. As a result, sometimes to converge to optimal solution quality, more number of rounds should be considered. In all of our experiments, h is considered in a way that the number of rounds r' not to increase too much.

According to the results which are depicted in Figures 3,4 and 5, parameter h should be chosen in way that in few number of messages and rounds, algorithm converges to a solution with high quality. Based on above discussions

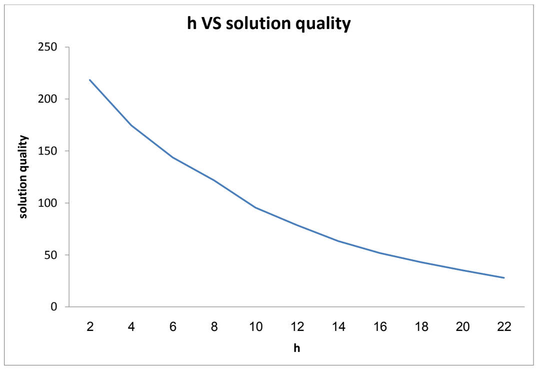


Figure 4. Solution quality is decreased by increasing h

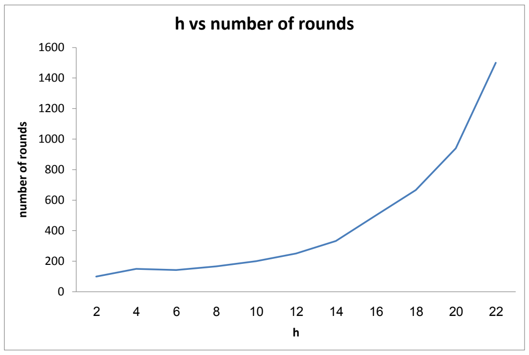


Figure 5. Total number of rounds are increased by increasing h

related to parameter h and number of rounds an estimation related to computational complexity of PKOPT is represented.

If we consider m the maximum number of messages which are passed in each group, total number of messages which are passed in each round of KOPT algorithm for the whole graph would be $m \times n$. Consequently, the total number of messages after r rounds would be:

$$\mathrm{T}_{KOPT} = n \times m \times r \tag{4}$$

In PKOPT algorithm this amount would be:

$$\mathrm{T}_{PKOPT} = \frac{n}{h} \times m \times r' \tag{5}$$

Where T is the total number of messages.

We introduce parameter β to compare computational complexity of KOPT and PKOPT with regard to parameter h.

$$\beta = \frac{\mathrm{T}_{PKOPT}}{\mathrm{T}_{KOPT}} \tag{6}$$

As we'll see in the next section and depending on the structure of a graph, the number of rounds in KOPT and PKOPT are not much different so if with a good approximation we ignore r and r' and consider $\frac{r}{r'} \approx 1$, by sequential approach, the number of messages passed among agents in each round decrease by $\frac{1}{h}$.

An example is given to show how the best value for h can be chosen. For a mesh graph with $n = 22$, different values for parameter h are chosen to reach a solution with the highest possible quality and low computational complexity. Figure 6 depicts β vs h. It is verified that h should be balanced according to the number of messages, the solution quality and the number of rounds. As it is clear, up to $h = 10$ in sequential PKOPT algorithm the computational complexity is decreased whereas the solution quality is almost the same as KOPT algorithm. For $h > 10$, number of rounds should be increased to reach the quality of KOPT solution.

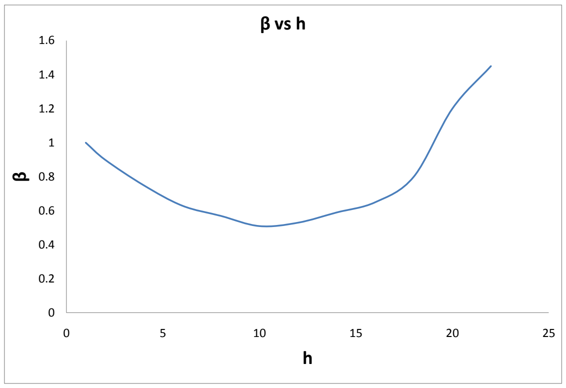


Figure 6. The best possible value for h is the minimum of the diagram.

For stochastic approach, we use normal distribution and set the leader to use threshold θ in range $[0,1]$. As mentioned in section IV each leader generates a number based on this function, then compares this number by its threshold to decide about activation. Due to many similarities we will not present the complexity analysis for the stochastic approach which is decreased to almost half in average.

VI. Experimental Result

In this section, We provide the experimental results of the proposed algorithms which are applied to mesh, fully connected and weakly chorded ring graphs. The results are shown for each of the above mentioned graphs with $n = 4, 6, 9, 18, 24$ where n is the number of agents. These structures are used to prove the efficiency of the proposed algorithms based on computational complexity and solution quality. Solution quality $\mathcal{R}$ is calculated based on equation 2 and computational complexity T is computed based on equation 4. These results confirm the better performance of the proposed algorithms in comparison with KOPT algorithm.

In all experiments the solution quality is considered as the stopping criterion and computational complexity of algorithms is utilized to compare the algorithms. To choose a proper value for parameter h, we do the same analysis described in section V and we decide to consider $h = 2$ for sequential PKOPT algorithm. In other words in each round only $\frac{1}{2}$ of leaders will be activated. For stochastic approach, we consider $\theta = \frac{1}{2}$ for each leader.

For mesh graph with $n = 4, 6, 9, 18, 24$, KOPT, stochastic PKOPT and sequential PKOPT algorithms are applied. Results are shown in Figure 7. The diagram on the top depicts solution quality based on these algorithms and the diagram on the bottom shows the computational complexity. For mesh graph with $n = 9$, the solution quality for stochastic PKOPT is $\mathcal{R} = 27$, for sequential PKOPT is $\mathcal{R} = 28$ and for KOPT algorithm is $\mathcal{R} = 29$. As it can be seen in Figure 7 the solution quality for different sizes of mesh graph does not have large discrepancy.

Computational complexity is another important factor in analyzing new algorithms. For mesh graph with $n = 9$, the computational complexity for KOPT is $\mathrm{T} = 2800$, for stochastic PKOPT is $\mathrm{T} = 2100$ and for sequential PKOPT algorithm is $\mathrm{T} = 1800$.

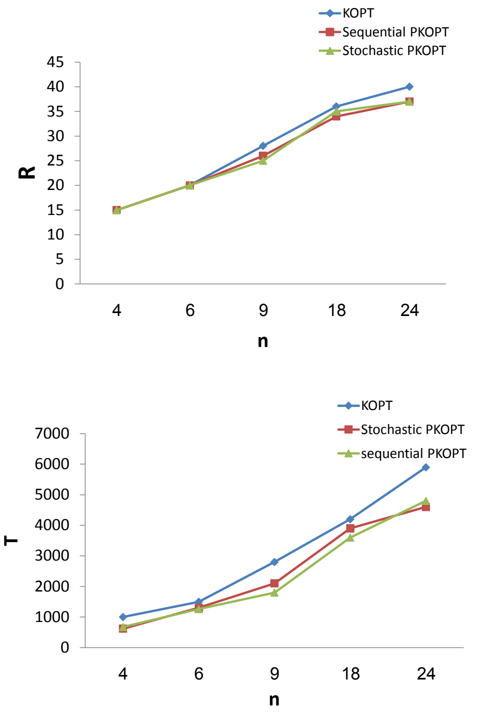


Figure 7. Mesh Graph

To have a complete evaluation of new algorithms both factors (solution quality and computational complexity) should be considered. For mesh graph with $n = 9$, the solution quality for sequential PKOPT is $\mathcal{R} = 28$ and for KOPT algorithm is $\mathcal{R} = 29$. Difference between the solution quality of these two algorithms is 1, as a result solution quality of KOPT is 3% better than the solution quality of sequential PKOPT . Difference between the solution quality of KOPT with $R = 29$ and stochastic PKOPT with $R = 27$ is 2 and as a result solution quality of KOPT 6% is better than the solution quality of stochastic PKOPT algorithms. On the other hand, computational complexity of KOPT and sequential PKOPT are $T = 2800$ and $T = 1800$ consecutively which shows that computational complexity is decreased 55% by sequential PKOPT algorithm. For KOPT, computational complexity is $T = 2800$ and for stochastic PKOPT is $T = 2100$ which shows that computational complexity is decreased 33%. It is obvious that the discrepancy of qualities is not too much and are tolerable than more computational cost.

Mesh and complete graphs are categorized as dense graphs. Consequently, the result for both graphs are very close to each other. As it is shown in Figure 8 the solution quality for all three types of algorithms has no discrepancy. In contrast, computational complexity decreases in a considerable amount. For graph with $n = 9$, computational complexity decreases 32% by sequential approach and 45% by stochastic approach.

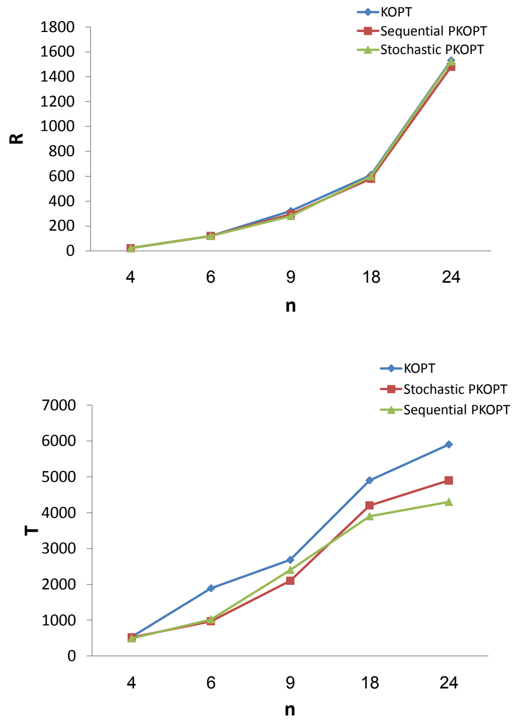
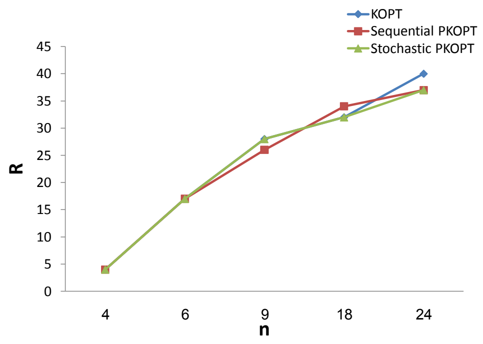


Figure 8. Complete Graph

Results of a weakly chorded ring graph are depicted in Figure 9. A weakly chorded ring graph is selected as a sparse graph to show the efficiency of algorithms for sparse graph in comparison with mesh graph which is a kind of dense graph. In graph with $n = 9$, utilities are close to each other for all three types of algorithms. On the other hand, according to the following discussions computational complexity is increased 5% by sequential PKOPT and 8% by stochastic PKOPT algorithm.



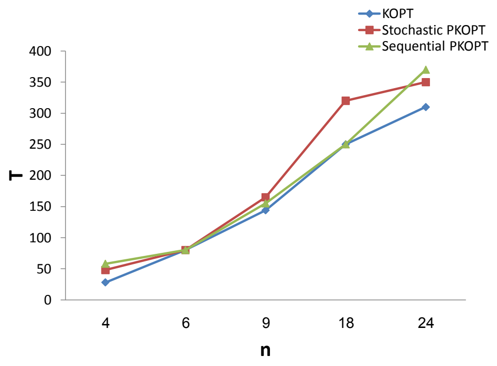


Figure 9. Weakly Chorded Ring Graph

There is a difference among the results for dense and sparse graphs based on two proposed approaches. In dense graphs, overlap among groups are more than the sparse graphs. Therefore, since the conflicts are being resolved number of leaders which set their assignment are not much and are close to the number of leaders which are activated in PKOPT algorithm. High number of overlaps means that groups have lots of agents in common and there is low discrepancy among quality of solutions of overlapping groups. Consider the graph in Figure 1, group 1 and group 2 have an overlap and the only difference is in agent 6. The best assignment for these groups are $\mathcal{R}_{\mathcal{G}_1} = 10$ and $\mathcal{R}_{\mathcal{G}_2} = 10$. By this simple example, it is clear that in dense graphs it is not very important which leaders are activated because the solution quality of groups are very close to each other. As a result, in dense graphs there is no difference among the solution quality of KOPT and two other proposed algorithms. Consequently, number of rounds do not change a lot.

In sparse graphs like weakly chorded ring graphs the number of overlaps among groups are not much. Thus, number of leaders which set their assignment in KOPT are more and the solution quality in KOPT will increase more than PKOPT in each round. In sequential and stochastic PKOPT algorithms, some leaders are activated which do not have the best solution quality among the others, therefore solution quality in each round does not increase as much as the solution quality of KOPT algorithm. As a result, number of rounds increase to converge to optimal solution quality in PKOPT. According to the description above, in sparse graphs, discrepancy in solution quality is more than dense graphs.

It can be concluded that the proposed algorithms have better performance in dense graph in comparison with sparse graphs.

VII. Conclusion and future work

In this paper, sequential PKOPT and stochastic PKOPT algorithms have been analyzed and it is shown that when the KOPT algorithm converges, sequential and stochastic PKOPT algorithms converge too. To be more precise on the result, the proposed algorithms were applied to different graphs. According to the results, the proposed algorithms for dense graphs will decrease the solution quality a bit, but will decrease the computational complexity in substantial amount. In sparse graphs such as weakly chorded ring graphs the results are a little different. In sparse graphs to reach an identical solution quality like KOPT algorithm, computational complexity will increase a bit. According to the discussion in this paper the proposed algorithms have better performance on dense graphs. The important issues which were investigated in this paper were balancing parameter h in sequential PKOPT and $\mathcal{P}$ in stochastic PKOPT algorithms. Based on experimental results, computational complexity decreases to $\frac{1}{h}$ in each round for a given h. Moreover, it was shown that by increasing parameter h the number of rounds will be increased. In stochastic PKOPT algorithm, parameter $\mathcal{P}$ has the same role and by increasing $\mathcal{P}$, computational complexity will be increased as well. Generally, depending the structure of the graphs, PKOPT algorithms reduces the number of messages and computations in each round, but may increase the time of convergence. In future works to improve the proposed algorithms, more investigation related to parameter h and $\mathcal{P}$ will be done and we try to find a new approach for group formation.

References

[1] H. J. Levesque, P. R. Cohen, and J. Nunes. On acting together. In AAAI, 1990.
[2] R. Mailler and V. Lesser. Solving distributed constraint optimization problems using cooperative mediation.

In 3rd International Joint Conference on Autonomous Agents and Multiagent Systems, pages $438-445$, July 2004.

[3] A. Petcu and B. Faltings. A Scalable Method for Multiagent Constraint Optimization. In 9th International Joint Conference on Artificial Intelligence, pages $266-271$, Aug. 2005.

[4] Maheswaran, R. T., E. Bowring, J. P. Pearce, P. Varakantham, M. Tambe, Taking DCOP to the real world: efficient complete solutions for distributed multi-event scheduling. Proceedings of the Third International Joint Conference on Autonomous Agents and Multi Agent Systems (AAMAS 2004). New York, NY, pp. $310-317$.

[5] P. J. Modi, W.-M. Shen, M. Tambe, and M. Yokoo. Adopt: Asynchronous distributed constraint optimization with quality guarantees. Artificial Intelligence, $16(12):149-180$, 2005.

[6] A. Chapman, R. A. Micillo, R. Kota, and N. Jennings. Decentralised dynamic task allocation: A practical game-theoretic approach. In The 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-09), pages $915-922, 2009$.

[7] S.M. Aji and R. J.McEliece. The generalized distributive law. IEEE Transactions on Information Theory, $46:325343, 2000$.

[8] M. Cooper, S. de Givry, and T. Schiex. Optimal soft arc consistency. In Proceedings of the 20th Internation Joint Conference on Artificial Intelligence (IJCAI07), pages $6873, 2007$.

[9] H. Katagishi and J. P. Pearce. KOPT: Distributed DCOP algorithm for arbitrary k-optima with monotonically increasing utility. In DCR-07, 2007.

[10] M. Yokoo and K. Hirayama. Distributed breakout algorithm for solving distributed constraint satisfaction and optimization problems. In Proc. ICMAS, pages 401408, 1996.

[11] S. Fitzpatrick and L. Meertens. Distributed coordination through anarchic optimization. In V. Lesser, C. L. Ortiz Jr., and M. Tambe, editors, Distributed Sensor Networks: A Multiagent Perspective, pages 257295. Kluwer Academic Publishers, 2003.

[12] R. T. Maheswaran, J. P. Pearce, and M. Tambe. Distributed algorithms for DCOP: A graphical-game-based approach. In PDCS, 2004.

[13] J. P. Pearce, M. Tambe,R. Maheswaran, Solving Multiagent Networks using Distributed Constraint Optimization, AAAI,2007.

[14] J. P. Pearce, M. Tambe. Quality Guarantees on k-Optimal Solutions for Distributed Constraint Optimization Problems, IJCAI 2007.

[15] J P. Pearce, R T. Maheswaran,Milind Tambe, Solution Sets in DCOPs and Graphical Games: Metrics and Bounds.

[16] Grimmett, G. R. and Stirzaker, D. R. Probability and Random Processes, 2nd Edition. Clarendon Press, Oxford.1992