

SOLVING DISTRIBUTED CONSTRAINT OPTIMIZATION PROBLEMS

An Evolutionary Approach

Maryam Rahmaninia, Elnaz Bigdeli

Institute for Advanced Studies in Basic Sciences, Zanjan, Iran

{m_rahmani, e_bigdeli}@iasbs.ac.ir

Mohsen Afsharchi

Department of Electrical and Computer Engineering, University of Zanjan, Zanjan, Iran

afsharchim@iasbs.ac.ir

Keywords: Multi agent systems, Distributed Constraint Optimization (DCOP), t -distance optimality.

Abstract: A significant body of work in multi-agent systems over more than two decades has focused on multi-agent coordination (Levesque et al., 1990). 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 the study of incomplete algorithms. In both complete and incomplete algorithms, computational cost is a major concern. Different approaches are introduced to solve this problem and improve existing algorithms. The main contribution of this paper is to decrease computational cost of DALO- t (Distributed Asynchronous Local Optimization) algorithm by introducing a new algorithm to find the best solution. This new algorithm is called Genetic Distributed Asynchronous Local Optimization (GDALO- t). GDALO- t is an effective method to reduce computational load and power consumption in implementation. This paper, under various assumptions, presents an analysis of this new algorithm.

1 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 impracticality 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 (Mailler and Lesser, 2004), (Petcu and Faltings, 2005). Distributed Constraint Optimization (DCOP) has been applied to

different domains. DCOPs are able to model the task of scheduling meetings in large organizations (Maheswaran et al., 2004). DCOPs are also able to model the task of allocating sensor nodes to targets in sensor networks (Modi et al., 2005). Finally, DCOPs are able to model the task of coordinating teams of unmanned vehicles in disaster response scenarios (Chapman et al., 2009).

There are two main categories for DCOP algorithms, complete and incomplete algorithms. Complete algorithms always find a configuration of variables that maximizes the global objective function. Adopt (Asynchronous Distributed OPTimization) (Modi et al., 2005) and DPOP (Dynamic Programming OPTimisation) (Petcu and Faltings, 2005) 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 (Aji and McEliece, 2000), Distributed Arc Consistency (Cooper et al., 2007) and KOPT (Katagishi and Pearce, 2007) 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 (Yokoo and Hirayama, 1996) and DSA1 (Fitzpatrick and Meertens, 2003) yield 1-optimal solutions. KOPT algorithm is the only incomplete algorithm which works for arbitrary k (Katagishi and Pearce, 2007). DALO- t is a novel asynchronous incomplete algorithm which works based on t -distance optimality is introduced in (Yin et al., 2009). In this algorithm each agent forms a group with other agents with a distance of t hops (Kiekintveld et al., 2010)(Yin et al., 2009).

In both KOPT and DALO- t algorithms, a complete DCOP algorithm is used to find the best value for each agent in a group. The computational complexity of complete algorithms is exponential in the number of variables n , since constraint optimization is known to be NP-hard (Modi et al., 2005). It is hard for complete algorithms to scale up because the computation burden might increase exponentially by increasing number of variables (Yin et al., 2009). Using t -distance optimality criterion to form groups, creates groups with large size in dense graph. Since, in each subgraph in DCOP a complete algorithm is applied, using a complete algorithm in each group is not tolerable from computational point of view. A new evolutionary algorithm is introduced in this paper which puts this problem right and finds the best solution in each subgraph in polynomial time.

The structure of the paper is as follows: In section 2, formal definitions of DCOP and t -distance optimality solutions are presented. In section 3, DALO- t algorithm and its main issues are described. The new algorithm is introduced in section 4. Experimental results of new algorithm and its comparison with DALO- t algorithm is depicted in section 5. Finally, conclusion and future work are presented in section 6.

2 BACKGROUND

In this section, we will provide some basic definitions about DCOP and t -distance optimality.

2.1 Distributed Constraint Optimization

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 $\mathcal{D} = \{\mathcal{D}_1, \dots, \mathcal{D}_n\}$ 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 $\mathcal{F}_{ij} : \mathcal{D}_i \times \mathcal{D}_j \rightarrow \mathfrak{R}$ 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 an input and returns a value as a non-negative number for a constraint.

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:

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

Where $v_i \leftarrow a_i, v_j \leftarrow a_j, a_i, a_j \in \mathcal{A}$ (1)

$\mathcal{R}(\mathcal{A})$ is a solution quality for a joint assignment \mathcal{A} (Pearce et al., 2007)(Modi et al., 2005).

Figure 1 shows an example DCOP with 5 variables and 6 constraints with identical cost function. The optimal assignment for this DCOP is $\mathcal{A} = (1, 1, 1, 1, 1)$.

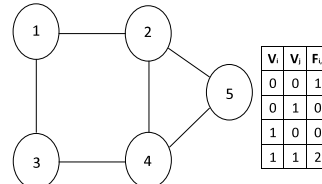


Figure 1: An example DCOP with five binary variables. Each constraint has the same cost function.

2.2 t -distance Optimality

Definition 1. For two different assignments \mathcal{A} and \mathcal{A}' :

$$\mathcal{D}(\mathcal{A}, \mathcal{A}') = \{v_i \in \mathcal{V} \mid a_i \neq a'_i, v_i \leftarrow a_i \in \mathcal{A}, v_i \leftarrow a'_i \in \mathcal{A}'\} \quad (2)$$

In other words, \mathcal{D} is a deviating group between two assignments \mathcal{A} and \mathcal{A}' .

Definition 2. For a pair of variables v_i and v_j , let $\mathcal{T}(v_i, v_j)$ be the shortest distance between them in the constraint graph. Let $\Phi_t(v_i) = \{v_j \mid \mathcal{T}(v_i, v_j) \leq t, v_i, v_j \in \mathcal{V}\}$ denotes a set of variables that can be reached from v_i within t hops.

Definition 3. A DCOP assignment \mathcal{A} is t -distance optimal if $\mathcal{R}(\mathcal{A}) \geq \mathcal{R}(\mathcal{A}')$ for all \mathcal{A}' , where $\mathcal{D}(\mathcal{A}, \mathcal{A}') \subseteq \Phi_t(v_i)$ for some $v_i \in \mathcal{V}$ (Kiekintveld et al., 2010)(Yin et al., 2009).

Example: Consider the graph in Figure 1. Given $t = 1$, 1-distance groups for variable 1 will be: $\Phi_1(v_1) = \{v_1, v_2, v_3\}$. In this example $\mathcal{A} = (0, 0, 0, 0, 0)$ with $\mathcal{R}(\mathcal{A}) = 6$ is 0-distance optimal. Assignment \mathcal{A} is 0-distance optimal because if every agent changes its value the solution quality will decrease. Assignment \mathcal{A} is 1-distance optimal too. This assignment is not 2-distance optimal because there is an assignment $\mathcal{A}' = (1, 1, 1, 1, 1)$ with $\mathcal{R}(\mathcal{A}') = 18$ which its utility is more than \mathcal{A} .

3 DALO- T ALGORITHM AND ISSUES

DALO- t algorithm, as an incomplete algorithm, was introduced by Yin. It is an asynchronous algorithm for DCOP based on t -distance optimality (Yin et al., 2009).

DALO- t algorithm has three phases. In phase one, each agent sends a message containing all its constraints to agents in a distance of t hops. Then, it broadcasts its initial value to a distance of $t + 1$ hops in a separate message. In phase two, based on information gathered in the previous phase, all the leaders compute a new optimal assignment by using a centralized variable elimination algorithm in parallel. In phase three, if the new assignment improves the solution quality, the group leader attempts to set the new assignment. When all the leaders try to set their assignments, there will be conflicts among overlapping groups, which are resolved by an asynchronous locking and commitment protocol.

DALO- t Issues Although DALO- t is an effective algorithm to solve DCOP problems, it suffers from some drawbacks. In t -distance optimality, the number of optimization groups is fixed, but the size of t -distance groups can be very large, particularly in dense graphs (Yin et al., 2009). Using distance as a criterion to create groups may produce groups with large number of nodes; especially, when there are hub nodes with many connections or subgraphs which are densely connected.

As it is explained in DALO- t algorithm in phase two, a complete algorithm is used to solve DCOP. A leader node uses a centralized variable elimination algorithm to solve the subproblem for the local group which is a complete algorithm. The computational complexity of complete algorithms is exponential in

the number of variables. On the other hand, by increasing t , the number of agents in a group will increase and using a complete DCOP solver will not be tolerated from size and space point of view. To solve this problem, instead of using a centralized variable elimination algorithm, we use a genetic approach in phase two. A DCOP is an optimization problem, and nowadays, evolutionary algorithms, particularly Genetic Algorithms (GAs), are considered as one of the best known algorithms to solve optimization problems.

4 ALGORITHM DESCRIPTION

In this section the overview of genetic algorithms and the GDALO- t algorithm are given.

4.1 The Genetic Algorithm Overview

Genetic Algorithms (GA) are adaptive methods and have been applied to optimization problems in many fields (Beasley et al., 1993). Algorithm is started with a set of solutions (represented by chromosomes) called population. Solutions from one population are taken and used to form a new population. This is motivated by a hope, that the new population will be better than the old one. Solutions which are selected to form new solutions (offspring) are selected according to their fitness - the more suitable they are the more chances they have to reproduce. This is repeated until some condition (for example number of populations or improvement of the best solution) is satisfied. Before a genetic algorithm is applied to a distributed constraint optimization problem, we show how chromosomes are represented and how the fitness of chromosomes are evaluated.

In this paper, each assignment for a group is considered as a chromosome. For group 1 in Figure 1, $A = (0, 1, 0)$ is a possible assignment and also a chromosome in genetic approach in which a_i indicates the value for the agent i . The fitness of a chromosome is evaluated by a fitness function. The fitness function which is used in this paper to evaluate the fitness of chromosome is the solution quality. Solutions with high quality are considered the fittest ones.

4.2 The Genetic Approach for DCOP

The GDALO- t algorithm is a new method to solve distributed constraint optimization problems. This algorithm has three phases.

4.2.1 Initialization

At first, every agent sends a message containing all its constraints to all agents which are in distance of t hops from it. Then, it chooses an initial value from its domain and broadcasts it to agents in distance of $t + 1$ hops. In this phase, a leader starts to construct its group. Given t , all the agents whose distance of center node are lower than t will be a member of the group. The additional hop in sending a message is for boundary nodes. The nodes in the boundary of a group are considered static in computing the best assignment.

4.2.2 Computing the Best Assignment

In this phase, the leader of each group calculates the best possible assignment by which the quality is maximized. In this phase, every leader utilizes genetic approach to find the best assignment. The genetic approach has four phases:

- **Initialization.** In this phase, an initial population is generated. This population is a set chromosomes (possible assignment for each group). The chromosomes are equal in size with the group and are a string of binary values. Generally, the population is generated randomly, and is selected from all possible solutions for a problem.
- **Selection.** To generate the next population, a set of solution is selected from existing population. Generally, the main criterion for selection is fitness and individuals which are high fit will be selected to generate the next generation. Using solution quality as a criterion a proportion of current population with high solution quality will be selected to generate the next population. However, the stochastic method is used in this paper. By using this method there will be a chance for less fit solutions.
- **Reproduction.** In this phase the next population is generated. Two chromosomes are selected to generate new chromosomes. Crossover and mutation are two basic operators of GA to generate new chromosomes for the next population. We use single point crossover in our implementation (Mitchell, 1998) and a mutation operator with a mutation probability of 0.007 to mutate values for an individual.
- **Chromosome Evaluation.** In this phase, newly generated offsprings are added to the population, then the worst individuals are removed from the population. Depending on whether they are better than the worst individuals in the population, the

new offsprings may, or may not, survive to join the new population. Mainly, some solutions with lower utility are selected since sometimes using these solutions to generate offsprings leads to high fit chromosomes.

Population generation is repeated until the algorithm converges to the optimal solution.

4.2.3 Implementing Assignments

Each agent belongs to different groups and receives various assignments from different leaders. To resolve the conflict among overlapping groups, a method is used for resolving conflict described in DALO- t algorithm (Yin et al., 2009).

5 EXPERIMENTAL RESULT

To evaluate the proposed algorithm, we use three different metrics. The first is the Number of Rounds (NR) which is the dominant metric for evaluation of DCOP algorithms and it is used in many papers (Yokoo et al., 1998), (Kiekintveld et al., 2010), (Davin and Modi, 2005). The second metric is Round Based Runtime (RBR) that measures the time required for computation and communication in each round (Davin and Modi, 2005). The third metric is the quality of solution which is introduced in section 2 and it is also a prominent metric to evaluate DCOP algorithms (Katagishi and Pearce, 2007), (Pearce et al., 2007). The main contribution of this work is to reduce the RBR. In the following, we provide the experimental results for the DALO- t and GDALO- t algorithms applied to graphs with different structures. Structure of graphs are chosen randomly and no special structure is used to prove the efficiency of the proposed algorithm for any structure. Our experiments use different graphs with sizes of $n = 30, 60, 80$. The density of graphs is considered $\mathcal{D} = \{0.11, 0.12, 0.14, 0.16, 0.2, 0.24, 0.33, 0.49\}$ and distance of t for creating groups is considered $t = 1, 2$. Each random graph used in the experiment is shown by a tuple (n, \mathcal{D}, t) . For each tuple (n, \mathcal{D}, t) , we create 5 different structures with size of n , density of \mathcal{D} , and distance of t . The results which are shown for each tuple are the average of the results for 5 different structures.

We use the same initial assignment for both algorithms and the number of rounds is considered same for two algorithms. The stopping criterion used to terminate the running of the algorithms is the number of rounds.

The main goal of this paper is to show that the required time to compute the best assignment is decreased by our evolutionary approach. To have a fair comparison, it should be proved that the other parameters do not change by using our new approach. The main concern is solution quality. It needs to be proved that by decreasing the time, the quality of solution will not change too much. It will be shown in the following section that the solution quality does not change very much by GDALO- t in comparison with DALO- t . As the experimental results show, GDALO- t decreases the computational cost and it can be concluded that our proposed algorithm is more efficient than DALO- t algorithm.

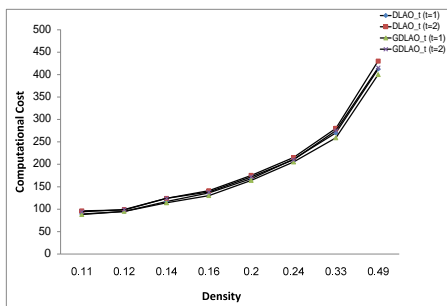


Figure 2: Quality of solution for graphs with size 30, different structures, densities and t s.

5.1 Solution Quality (SQ)

To compare the algorithms, the quality of solution for DALO- t and GDALO- t are evaluated. Figure 2 depicts the results for $n = 30$. As it is shown in the results, a little difference is observed between quality of solutions of GDALO- t and DALO- t algorithms. The difference among solutions quality is various for different graphs; in some cases, the difference is 2%, in some graphs is 3%, but the difference does not exceed 5%. On average, the difference among the solution quality of these two algorithms are 4.42%; therefore, it can be ignored. For example, for the tuple $(30, 0.2, 1)$, the solution quality is 168 for DALO- t and is 164 for GDALO- t . These results show that GDALO- t algorithm finds the optimal solution like DALO- t algorithm in many cases.

5.2 The Computational Cost (CC)

As it is described RBR is used to compare computational cost. The results for graph with $n = 80$ is depicted in Figure 3. For example, for the tuple $(80, 0.33, 2)$, the elapsed CPU time is 4700 seconds for DALO- t whereas this parameter for GDALO- t algorithm is only 3153 seconds. It is obvious that using

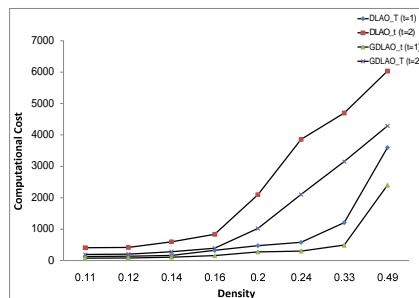


Figure 3: Computational cost to solve DCOP for graphs with size 80, different structures, densities and t .

GDALO- t , the computational cost decreases in considerable amount. As the size of active agents in a group increases, the elapsed CPU time of DALO- t algorithm increases exponentially, but using GDALO- t algorithm, the time increases polynomially. The difference between the result of running two algorithms for $t = 1$ and $t = 2$ makes the efficiency of the algorithm more clear. In the graph with density of 0.2 the size of groups is more than graphs with density of 0.11, and computing all the possible assignments and finding the best assignment will be complicated in dense graphs. The problem will be more complicated considering $t = 2$ because by increasing t the size of groups in the graph will increase, and consequently, the number of possible assignments increases. Obviously, the difference between computational cost of DALO- t and GDALO- t for a graph with the tuple $(80, 0.2, 2)$ is more than a graph with the tuple $(80, 0.11, 1)$ as it is depicted in Figure 3.

The other important point which is clear in the results is that in dense graphs the proposed algorithm is more efficient. As it is shown in Figure 3 for the graph with size $n = 80$ and $t = 1$ and density $\mathcal{D} = 0.33$ computational cost is decreased by 59% and for the the same graph with density $\mathcal{D} = 0.11$ it is decreased by 38%.

5.3 Computational Cost vs Solution Quality

To have a complete comparison between DALO- t and GDALO- t algorithms, computational cost and solution quality should be used together in the comparison. The results for different tuples are shown in Table 1. The average difference is shown for both algorithms. The quality of GDALO- t algorithm solutions is 4.42% lower than DALO- t algorithm solutions on average. On the other hand, the computational cost of GDALO- t algorithm is 43% lower than DALO- t algorithm on average.

Table 1: Results for different tuples using DALO- t and GDALO- t .

Graph (n, \mathcal{D}, t)	DALO- t algorithm (SQ, CC)	GDALO- t algorithm (SQ, CC)
(30,0.11,1)	(89,15)	(88,9)
(30,0.2,1)	(168,42)	(164,29)
(30,0.24,2)	(215,340)	(210,220)
(30,0.49,1)	(412,310)	(400,162)
(60,0.12,1)	(428,81)	(411,24)
(60,0.2,1)	(670,170)	(650,89)
(60,0.33,1)	(1130,590)	(1110,279)
(60,0.49,2)	(1765,3800)	(1600,2300)
(80,0.14,1)	(855,166)	(830,109)
(80,0.2,1)	(1220,480)	(1170,279)
(80,0.33,1)	(2050,1209)	(2050,494)
(80,0.33,2)	(2098,4700)	(1900,3153)
Average Discrepancy	(0.0442,0.434)	

Obviously, computational cost has direct influence on quality of solution. By decreasing the computational cost, the quality of solution will decrease. Therefore, we should find a balance between the quality of solution and the cost consumed to reach to a special quality.

6 CONCLUSIONS AND FUTURE WORK

In this paper, instead of using a complete algorithm in each group in DCOP, a genetic algorithm is used. As it is shown in the results, the quality of solutions achieved by complete algorithm is more than genetic algorithm. The quality of solutions achieved by genetic algorithm is 4.42% lower than DALO- t algorithm on average. This result is predictable because complete algorithms find the optimum solutions while the genetic algorithm finds the semi-optimal solutions. In contrast, the computational cost decreases considerably by using genetic approach. The computational cost decreases 43% by using GDALO- t . It can be concluded that our algorithm is more applicable for real-time applications. Decreasing the number of messages passed to solve DCOP is a key avenue for future work.

REFERENCES

- Aji, S. and McEliece, R. (2000). The generalized distributive law. In *IEEE Transactions on Information Theory*, 46 : 3253-343. IEEE.
- Beasley, D., Bull, D., and Martin, R. (1993). An overview of genetic algorithms: Part 1, fundamentals. University of Cardiff.
- Chapman, A., Micillo, R., Kota, R., and Jennings, N. (2009). Decentralised dynamic task allocation: A practical game-theoretic approach. In *In Proc. of AAMAS-09*, pages 915 – 922. IFAAMAS.
- Cooper, M., Givry, S., and Schiex, T. (2007). Optimal soft arc consistency. In *In Proc. of IJCAI07*, pages 6873. Morgan Kaufmann Publishers Inc.
- Davin, J. and Modi, P. (2005). Impact of problem centralization in distributed constraint optimization algorithms. In *In DCR*. ACM New York, NY, USA.
- Fitzpatrick, S. and Meertens, L. (2003). Distributed coordination through anarchic optimization. In *Distributed Sensor Networks: A Multiagent Perspective*, pages 257-295. Kluwer Academic.
- Katagishi, H. and Pearce, J. (2007). Kopt: Distributed dcopt algorithm for arbitrary k-optima with monotonically increasing utility. In *In DCR-07*.
- Kiekintveld, C., Z. Yin, A. K., and Tambe, M. (2010). Asynchronous algorithms for approximate distributed constraint optimization with quality bounds. In *Proc. of 9th Int. Conf. on AAMAS10*. AAMAS.
- Levesque, H., Cohen, P., and Nunes, J. (1990). On acting together. In *In Proc. of the National Conference on Artificial Intelligence*. IAAA.
- Maheswaran, R., Bowring, E., Pearce, J., Varakantham, P., and Tambe, M. (2004). Taking dcopt to the real world: efficient complete solutions for distributed multi-event scheduling. In *In Proc. of AAMAS04*.
- Mailler, R. and Lesser, V. (2004). Solving distributed constraint optimization problems using cooperative mediation. In *In Proc. of AAMAS04*.
- Mitchell, M. (1998). *An introduction to genetic algorithms*. MIT Press Cambridge, MA, USA.
- Modi, P., Shen, W., Tambe, M., and Yokoo, M. (2005). Adopt: Asynchronous distributed constraint optimization with quality guarantees. In *Artificial Intelligence*, 16(12) : 149 – 180. Elsevier, Oxford, ROYAUME-UNI.
- Pearce, J., Tambe, M., and Maheswaran, R. (2007). Solving multiagent networks using distributed constraint optimization. In *AI Magazine*. AAAI.
- Petcu, A. and Faltings, B. (2005). A scalable method for multiagent constraint optimization. In *In Proc. of IJCAI05*. Morgan Kaufmann Publishers Inc. San Francisco, CA, USA.
- Yin, Z., Kiekintveld, C., Kumar, A., and Tambe, M. (2009). Local optimal solutions for dcopt: New criteria, bound, and algorithm. In *Proc. of 8th Int. Conf. on AAMAS09*. AAMAS.
- Yokoo, M., E.H. Durfee, Ishida, T., and Kuwabara, K. (1998). The distributed constraint satisfaction problem: Formalization and algorithms. In *Knowledge and Data Engineering*, 10(5):673-685.
- Yokoo, M. and Hirayama, K. (1996). Distributed breakout algorithm for solving distributed constraint satisfaction and optimization problems. In *In Proc. ICMAS*, pages 401-408.

This document was created with Win2PDF available at <http://www.daneprairie.com>.
The unregistered version of Win2PDF is for evaluation or non-commercial use only.