Evaluating the Performance of DCOP Algorithms in a Real World, Dynamic Problem

Robert Junges and Ana L. C. Bazzan*
Instituto de Informática, UFRGS
Caixa Postal 15064
91501-970 P. Alegre, Brazil
{junges,bazzan}@inf.ufrgs.br

ABSTRACT

Complete algorithms have been proposed to solve problems modelled as distributed constraint optimization (DCOP). However, there are only few attempts to address real world scenarios using this formalism, mainly because of the complexity associated with those algorithms. In the present work we compare three complete algorithms for DCOP, aiming at studying how they perform in complex and dynamic scenarios of increasing sizes. In order to assess their performance we measure not only standard quantities such as number of cycles to arrive to a solution, size and quantity of exchanged messages, but also computing time and quality of the solution which is related to the particular domain we use. This study can shed light in the issues of how the algorithms perform when applied to problems other than those reported in the literature (graph coloring, meeting scheduling, and distributed sensor network).

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence
Multiagent Systems, Coherence and Coordination, Intelligent Agents

General Terms

Algorithms

Keywords

Coordination, Traffic Control, Distributed Constraint Optimization

1. INTRODUCTION AND MOTIVATION

It is generally accepted that the formalism underlying distributed constraint optimization problems (DCOP) represents a generic framework for the resolution of distributed problems in multiagent systems (MAS), especially in problems where the challenge is to find the best value attribution for a set of variables that have interdependencies. However there is also a general perception that complete (i.e. not approximate) algorithms proposed to solve DCOP's have problems with efficiency [8]. Although this has not prevented the use of these algorithms in real world scenarios, comparison of methods has been restricted to a few problems such as graph coloring, meeting scheduling, and distributed sensor network. There is a lack of studies about how to employ DCOP in other classes of problems, especially in dynamic changing environments. There, it is not obvious how the DCOP formalism performs.

In a DCOP, differently from a distributed constraint satisfaction problem (DisCSP) [20], the interest is to optimize the restrictions and not only to satisfy them. DCOP is represented by a global function and the objective is to maximize or minimize it. This function depends on a cost value associated with each restriction.

Approaches for dealing with DCOP in real life problems should consider that the agents must be able to optimize the global function in a distributed way, using only local communication. A DCOP algorithm should also be capable of finding the solution with the agents working in an asynchronous way, and provide quality guarantees. Besides these intrinsic characteristics of the algorithms, these must also be evaluated regarding the quality of the solution in the particular scenario or problem where it is applied.

In [8] a real world application of DCOP is discussed in which, while using complete algorithms (in this case ADOPT), heuristics for better communication structure and precomputing best case bounds were necessary. Authors report that they have encountered fundamental differences between abstract scenarios (e.g. graph coloring), and concrete ones (meeting scheduling, sensor networks). This has enabled them to handle scenarios with up to 47 variables (roughly agents).

Later, in [15] this was extended to more than hundred variables with the use of another kind of DCOP algorithms, this time based on dynamic programming. This algorithm, DPOP, is linear in the number of agents. The disadvantage is that performance is achieved with a high increase in message size and memory (the growth is exponential in the width of the tree).

Both [8, 15] concentrate on the the analysis of the computational efficiency, but less on their efficiency and effectiveness regarding the real world problem they are addressing.

In the present paper we test and compare three DCOP algorithms in a problem of the domain of traffic, namely synchronization of traffic lights. This problem can be seen as an assignment problem because the objective is to find the best signal plan for each traffic light in order to maintain a progressive system (see Section 3 for an explanation). This assignment is challenging because the progression cannot be done in more than one traffic direction for each traffic light. Therefore each must select a signal plan that best matches the traffic load not only locally but also in the neighborhood. Moreover, it is a dynamically changing environment.

The algorithms we compare are *Asynchronous Distributed Optimization* (ADOPT) [10], *Optimal Asynchronous Partial Overlay* (OptAPO) [9], and *Dynamic Parameter Optimization Problem*

*Author partially supported by CNPq

Cite as: Evaluating the Performance of DCOP Algorithms in a Real World, Dynamic Problem, R. Junges and Ana L. C. Bazzan, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons (eds.), May, 12-16., 2008, Estoril, Portugal, pp. 599-606.

(DPOP) [15].

In the next section the DCOP framework is briefly presented; Section 3 explains the domain, the problem of synchronization (progressive system), and describes the scenario. The experiments and their analysis appear in Section 4. Section 5 reports the conclusions and future work.

2. DISTRIBUTED CONSTRAINT OPTIMIZATION PROBLEMS

In the last decade, several DCOP algorithms were proposed; here we restrict the analysis to three complete ones: ADOPT, DPOP, and OptAPO. Being complete, they must guarantee an optimum solution. Another characteristic is that in these algorithms, agents are asynchronous (execute at the same time).

ADOPT performs a distributed search using cost communication as a guide for the agents to choose the optimum values for its variables. OptAPO uses direct constraint communication as a form of partially centralize the problem, through a mediator. The choice of the mediator is done during the resolution process using priorities given to the agents. The mediator uses a centralized optimization in order to find an optimum solution to its portion of the problem. DPOP is based on dynamic programming, propagating utility in a tree-like network of agents. The growth of the number of messages is linear; messages' size is domain-specific.

The solution of distributed constraint optimization problems requires a set of agents that use communication to find the best value attribution to their variables. Two agents sharing a restriction are called neighbors. A DCOP can be formalized as follows:

- a set of n agents, $A = \{a_1, a_2, ..., a_n\}$;
- a set of n variables, $V = \{v_1, v_2, ..., v_n\}$;
- a set of domains $D = \{d_1, d_2, ..., d_n\}$, where $v_i \in d_i$. Each d_i is finite and discrete;
- a distributed mapping $Q_l : v_i \rightarrow a_j$ assigning each variable to an agent. $Q(v_i) = a_j$ means that the agent a_j is responsible for the variable v_i.
- a set of cost functions $f_{i,j} : d_i \times d_j \rightarrow \mathbb{N}$, to the pair of variables v_i and v_j. Cost functions are also called restrictions;
- an objective function F, defined as an aggregation over the set of restrictions. The objective is to find the set of values O^* for the variables V, minimizing or maximizing the objective function. The function F is defined as $F(O) = \sum f_{i,j}(v_i, v_j)$, where $v_i \leftarrow d_i, v_j \leftarrow d_j$ in O.

In the next section we give details about how to use this formalism in the particular scenario we tackle here.

3. TAKING DCOP TO (ANOTHER) REAL WORLD PROBLEM: A CASE STUDY ON SYNCHRONIZATION OF TRAFFIC LIGHTS

3.1 Motivation

A classical approach to reduce traffic jams is to maintain a so-called progressive system, i.e. to coordinate or synchronize traffic lights so that vehicles can traverse an arterial in one traffic direction, keeping a specific speed, without stopping. Classical approaches to building a progressive system are [6, 19]. Thus, coordination

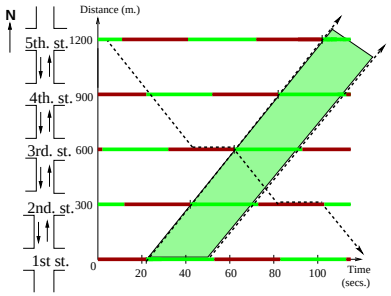

Figure 1: Time-space diagram of a progression in an arterial

here means that if appropriate signal plans are selected to run at the adjacent traffic lights, a "green wave" is built so that drivers do not have to stop at junctions. Details are given in Section 3.2).

To solve this kind of problem, we are interested in achieving a compromise between totally autonomous coordination without explicit communication, and classical solutions which are either centralized or do not deal with network optimization. Here we approach the problem using DCOP algorithms. It is important to remember that these algorithms were not primarily designed for dynamic environments. However, traffic control scenarios are intrinsically dynamic. The ultimate goal is to design protocols for traffic light control to work without intervention of traffic experts.

The specific objective of the scenario is to find the best signal plan configuration for the traffic lights. The configuration is achieved by selecting the best signal plan for the current traffic situation constrained by the synchronization possibilities. The traffic models are implemented and their results evaluated using IT-SUMO, an agent-based traffic simulation framework [18].

3.2 Progressive Systems: Traffic Light Synchronization

Control of traffic lights is turning more important given the increasing demand for urban mobility. Since it is not always feasible to increase the existing network, one option is to efficiently use this infrastructure. Thus, intelligent control systems are necessary in order to increase the automation of the traffic control [14]. Traffic lights are normally operated based on pre-established traffic signal plans which are designed to work with given nominal volume of vehicles. However this is not flexible and cannot cope with changes in these volumes.

The aim of the use of DCOP in this scenario is to assign a traffic signal plan (henceforth signal plan for short) to each traffic light in order to guarantee synchronization of these lights. Synchronization means that adjacent traffic lights will be coordinated so that vehicles departing at traffic light A (upstream) will be able to pass traffic light B (downstream) without stopping. This is achieved by a synchronization of the plans which differ only by an offset (time between the beginning of the green phase of two consecutive traffic signals). Of course this offset is a function of the distance between traffic lights and the desired (average) speed of vehicles.

Figure 1 shows a space-time diagram of the synchronized or progressive system in an arterial with 5 crossings (depicted as the left side of the diagram). The progression is effective for vehicles traveling from the First to the Fifth Street (northbound). One bandwidth is shown: vehicles entering the arterial in the intersection with the First Street from time 22 to 54 will be able to pass the whole arterial without stop. Notice that this is not the case regarding a vehicle traveling southbound (shown in the figure) or west and eastbound (not shown).

A traffic light has a library of plans, each allowing the synchronization in a different traffic direction. In general, the more neighbors that are synchronized, the shorter the queues. Synchronization in two opposing directions of an arterial can be achieved depending on the geometry of the arterial: in a Manhattan-like grid where the spacing among intersections is the same, synchronization in opposite directions *in the same street* is possible but demanding. Synchronization in four traffic directions is not possible in practice. Therefore, an agent at a junction must *select* a signal plan to give priority to the synchronization in a particular traffic direction.

In order to explain our approach we use Manhattan-like grids. However, for the sake of generalization, the plans we have designed do not allow synchronization in more than one traffic direction in the same street.

3.3 Related Work on Progressive Systems

Classical algorithms were proposed in the sixties and seventies to analyze traffic patterns and to set synchronization in arterials. The most known is TRANSYT, an *off-line* optimization tool [19]. SCOOT [6] and SCATS [7] work similarly to TRANSYT but are traffic responsive. A new approach is TUC [4], conceived for large scale networks. Authors report positive results compared to a situation with fixed time synchronization. On the other hand, in TUC the computation is centralized and the conflits are solved either by a traffic expert or by a priori rules, in a clear contrast with a trend of decentralization of control.

It is important to note that none of these approaches deals with optimization at the level of grid. SCOOT and SCATS work predominantely at the level of crossings. TRANSYT is able to optimize an arterial (not necessarily one single, straight street) but does not deal with the kind of grid we use in the present paper. In TRANSYT, crossings that form a progressive system must be defined in advance, whereas in our case the groups of coordination emerge from the optimization process. TUC, as mentioned before, deals with networks but the conflicts (e.g. which progression group each crossing belongs) must be solved manually.

In [1] a MAS based approach is described in which each traffic light has a set of pre-defined signal plans to coordinate with neighbors. This approach uses techniques of evolutionary game theory: self-interested agents receive a reward or a penalty given by the environment. The benefits of this approach are threefold: it is not necessary to have a central agent to determine the direction of the coordination; agents can dynamically build subgroups of traffic light coordination which meet their current needs in terms of allowing vehicles to pass in one given direction; it avoids communication between agents when they have to decide in which direction to coordinate. However, payoff matrices (or at least the utilities and preferences of the agents) are required, i.e these figures have to be explicitly formalized by the designer of the system. This makes the approach time-consuming when many different options of coordination are possible and/or the traffic network is complex.

In [13] an approach based on swarm intelligence is proposed. There, each intersection (plus its traffic lights) behaves like a social insect that grounds its decision-making on mass recruitment mechanisms found in social insects. Signal plans are seen as tasks to be performed by the insect without any centralized control or task allocation mechanism. Stimuli to perform a task or to change tasks, are provided by the vehicles that, while waiting for their next green phase, continuously produce some "pheromone". No other information is available to the agents. Quantitatively, when the agents are free to decide coordinating according to the swarm approach, the system behaves almost as a central decision support system. Experiments show that the agents achieve synchronization without any central management. However, the time needed to converge to a stable coordination can be high, which is a negative aspect especially in highly dynamic environments.

Another approach based on an algorithm for DCOP is proposed in [12]. OptAPO was employed and tested in an environment that changes dynamically. One first shortcoming of the mediation-based method is that the mediation may end up being performed by a single agent, thus in a centralized way. Additionally, the mediation process may take time, as it was reported in [12] for a network with 25 agents.

In summary, there are classical and AI and/or multiagent based approaches to traffic light synchronization. Besides these, other approaches – not for synchronization – were proposed, e.g. based on reinforcement learning [3, 11, 17] or crossings with no lights [5].

3.4 DCOP Model

The problem of traffic light synchronization or coordination, formalized as a DCOP, is described by the tuple $< A, D, F >$ where:

- $A = \{a_1, ..., a_n\}$ is the set of variables/agents, where n is the number of crossings;
- $D = \{d_1, ..., d_n\}$ is the domain of the variables, representing the possible signal plans for each crossing agent;
- $F = \{f_{1,1}, ..., f_{n,n}\}$ is the set of constraints among the variables, where constraint f_i denotes the associated cost; each constraint has a cost for a given pair of values of the two variables.

As in the majority of DCOPs, constraints are binary i.e. they occur between two traffic light agents. Be $\rho_{i,j}$ the density of vehicles in the lane $i \rightarrow j$ (how heavy is the load). We then define $\beta_{i,j}$ to represent the fraction of traffic at intersection j that is coming from direction i:
$\beta_{i,j} = \frac{\rho_{i,j}}{\sum_z \rho_{z,i}}$
where z is the set of all nodes which send vehicles to node j.

Further, γ expresses the degree two consecutive agents agree (are synchronized); the value of 1.5 is used when agents are synchronized; otherwise we use 2. Another quantity, τ, expresses the degree agents are coping with the volume of vehicles. For example, if β indicates a higher volume of vehicles westwards and the agent is synchronizing in this direction, then the agent is executing its best plan (it is synchronized in the direction of higher volume) and has the lowest cost. τ assumes different values. Table 1 shows all values of τ, for different traffic and synchronization conditions for lanes between crossing *i* and crossing *j*.

Costs are given by the volume of vehicles that use the existing lanes between the traffic lights, and also by the degree to which these lights are synchronized. Thus the cost $f_{i,j}$ associated with a constraint between two neighboring crossings i and j is:

$$f_{i,j} = \beta_{i,j} \times \tau \times \gamma$$

Plan run by agent i	Plan run by agent j	τ
$\oplus$	$\oplus$	0
$\ominus$	$\oplus$	1
$\oplus$	$\ominus$	1.5
$\ominus$	$\ominus$	2

Table 1: τ for different configurations of plans and traffic volume. $\oplus$ means that plan is synchronized and agrees with the direction of higher traffic volume; $\ominus$ means that plan is synchronized in a direction other than that of higher traffic volume

3.5 Scenario Description

For the experiments reported here, we use four different scenarios or grids with different sizes (3x3, 5x5, 7x7, and 9x9). Thus the parameter n is set to 9, 25, 49, and 81.

All networks have source nodes that insert vehicles at a given rate. In the case of the 5x5 network, there are 10 such sources, one for each street. Each street is one-way. However, vehicles can turn in each crossing with a given probability. Also, vehicles can decelerate with a given probability. The higher these probabilities, the more traffic there is at any given instant. Each crossing has a set of signal plans defined in Section 4.1.

3.6 Metrics for Evaluation

We perform two types of comparison: evaluation of the performance of the algorithms themselves, and the quality of the control they achieve in this particular domain of traffic lights synchronization. In order to evaluate the former we measure: the number of cycles (as in other works, one DCOP cycle is defined as all agents receiving all incoming messages and sending all outgoing messages); execution time; average size of the messages exchanged among the agents; and the total number of messages exchanged.

To evaluate the DCOP based control we measure averages of total number of stopped vehicles and density (vehicles per link, normalized between 0 and 1). In all cases the averages are over all lanes of the network.

4. EXPERIMENTS AND RESULTS

All simulations were performed in the ITSUMO traffic simulator, running in an Intel Centrino Core Duo with 1.6GHz, 512MB RAM memory, under Suse Linux 10.2. Values reported here are averages and standard deviations over at least 10 repetitions of the simulation. Control actions happen once each 720 time steps (12 minutes) as it is not common practice in traffic engineering to change the signal plan within small time frames.

4.1 Signal Plans

Signal plans were designed to allow synchronization in either the NS/SN direction or in the EW/WE directions (remember lanes are one way). There are two phases, each allowing 30 seconds of green time to a given direction. Thus the traffic light cycle time is 60 seconds.

Domains $d_i = \{1, 2\}$, where 1 means EW/WE direction and 2 means NS/SN direction. As said, synchronized plans in neighboring intersections differ only by an offset. Given the speed of 15 m/s and the distance of 300 meters, the offset is 20 seconds.

4.2 Changing the Volume of Traffic

We consider situations with both dynamic and static flow of vehicle in the network (situations 1 and 2 as defined below). However, "static" does not mean that queues are deterministic as there are two sources of stochastic behavior in the simulation: turning probabilities at each crossing, and deceleration probabilities (each vehicle). Therefore, the word static here means only that the insertion rate of vehicles is constant.

1. **Situation 1 - Sources with Constant Insertion Rate**: in this case all sources generate vehicles with a fix rate of 0.4 during 9000 time steps (2.5 hours);

2. **Situation 2 - Sources with Changing Insertion Rate**: in this case there are changes in the insertion rate each 5000 steps, according to Table 2.

4.3 Case Studies

In addition to the two situations where we change the insertion rate or keep it constant, we also run different case studies concerning what kind of signal plans agents use. We summarize this below:

- **Case 1 (signal plans are not synchronized)**: we set signal plans at each agent to mismatch completely those running at neighbors, aiming at a quite bad configuration (a kind of lower bound);

- **Case 2 (signal plans are synchronized in a pre-defined direction)**: here we use the EW/WE, aiming at testing a common situation when signal plans are designed to cope with a specific pattern but changes occur in an unexpected way;

- **Case 3 (use of DCOP algorithms to perform the control)**.

DCOP algorithms are: ADOPT from `http://teamcore.usc.edu/dcop/`; OptApo implemented according to [9]; and DPOP from `http://liawww.epfl.ch/frodo/`.

4.4 Evaluation of the Algorithms

In this subsection we present the evaluation of the computational performance of the algorithms in terms of the standard measures (number of DCOP cycles, messages) and also include running time. The evaluation in terms of the efficiency at controlling the traffic is presented in the next subsection. In any case we use the scenarios and DCOP modeling described in sections 3.4 and 3.5, as well as the signal plans, traffic volumes and case studies described above.

Tables 3 to 7 show the averages over the whole simulation time and over 10 repetitions of the simulation. These numbers refer to Situation 2 i.e. with changing in insertion rate. Table 3 refers to execution time and number of cycles for the three DCOP algorithms used. Tables 4 to 7 then show the number of messages exchanged and the size of these messages.

For all three algorithms, the execution time was adequate in the sense that this time is below the time the agent at the crossing performs a control action (as said, 720 time steps). In other words, the running times achieved are compatible with the decision making frequency. The only exception is ADOPT when the grid is 9x9. In this case, the time necessary to reach a solution was almost 10 times higher than the control period. Apart from this exception, we can conclude that, even if the agent control frequency would have to be higher (and hence the time between two interventions smaller), there would still be some room for agents to run a DCOP assignment since the time to find a solution is below the time a signal plan runs a complete cycle through all phases of the traffic light (60 steps) in the following cases: network 3x3 and 5x5 (all algorithms), OptAPO and DPOP in the network 7x7.

A note about DPOP is that the number of cycles and messages is linear in the number of agents and depends on the height of the tree associated with the problem being modelled. According to

ID	Time Frame	Rate NS/SN	Rate EW/WE
Before 6 a.m.	0 to 5000	0.10	0.10
Morning	5000 to 10000	0.10	0.40
Afternoon	10000 a 15000	0.40	0.10
Rush Hour I	15000 a 25000	0.30	0.30
Rush Hour II	25000 a 30000	0.50	0.50
Night	30000 to 35000	0.10	0.10

Table 2: Changes in insertion rates at sources (situation 2)

Net.	Alg.	Time (sec.)	Cycles
3x3	ADOPT	0.65 ± 0.15	2186.13 ± 648.03
	OptAPO	0.16 ± 0.39	29.08 ± 23.20
	DPOP	0.07 ± 0.01	18
5x5	ADOPT	16.07 ± 2.29	6197.81 ± 1520.67
	OptAPO	3.90 ± 1.11	56.54 ± 15.76
	DPOP	0.22 ± 0.02	50
7x7	ADOPT	112.90 ± 33.39	12387.84 ± 3098.56
	OptAPO	21.16 ± 17.53	75.33 ± 14.02
	DPOP	13.59 ±0.01	98
9x9	ADOPT	6727.46 ±182.98	325512.42 ±10108.67
	OptAPO	295.75 ±31.56	174.30 ±18.41
	DPOP	180.71 ±15.35	152.00

Table 3: Average time and number of cycles (situation 2)

[15], the number of cycles is two times the height of the pseudotree (one propagation of the "util" message, and one propagation of the "value" message). Since the tree does not change from one repetition to the other of the simulations, the deviation is zero.

OptAPO can be seen as a good compromise in terms of execution time and communication (message exchange). However, for bigger networks, where there is a huge number of restrictions and costs are higher due to the increase on the insertion rates, the mediation process becomes more frequent. This mediation means that there is a partial centralization, which can be an issue as we intend to have a distributed solution. In order to address this, the approach proposed in [2] could be tried. Another issue is that our implementation of OptAPO is not fully distributed. If it were, we would see an increase on the number of message exchanged and that would affect the execution time.

For a better comparison of numbers regarding the execution time of all algorithms in the four scenarios, we show these in Figure 2.

Regarding the number and size of the exchanged messages (see Figure 3), as mentioned, DPOP is linear in the number of agents. Thus, in general, the DPOP algorithm performed better in our tests in terms of number of exchanged messages (also regarding execution time and number of cycles). However, the same cannot be said about the size of the messages, which are much higher, especially when the number of agents increases, see Figure 4. This happens because each message has more information, aggregated from the communication between parents and children in the tree. In real world applications, it is generally the case that there is a limitation in the communication bandwidth, and this can be an issue.

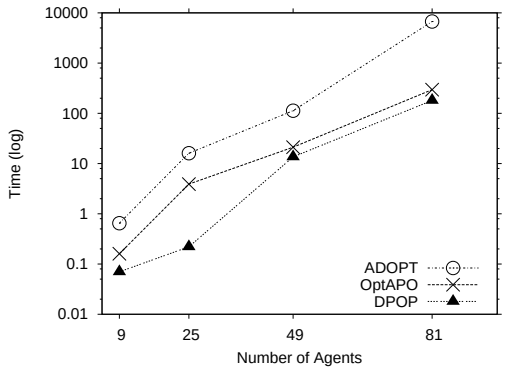

Figure 2: Running times (log scale) for the three algorithms, for 9, 25, 49 and 81 agents (Situation 2).

Alg.	Nb. of Msgs	Msgs size (bytes)	Total (MB)
ADOPT	298.25 ± 63.37	47.39 ±14.90	13.8
OptAPO	203.58 ± 153.05	6.80 ±1.98	1.35
DPOP	36	55.54 ± 2.66	1.95

Table 4: Network 3x3: average number and size of messages

Alg.	Nb. of Msgs	Msgs size (bytes)	Total (MB)
ADOPT	4123.70 ± 2130.45	66.38 ±13.84	231
OptAPO	766.36 ± 141.46	25.22 ± 6.44	19
DPOP	112	295.10 ± 3.80	33

Table 5: Network 5x5: average number and size of messages

Alg.	Nb. of Msgs	Msgs size (bytes)	Total (MB)
ADOPT	15972.9 ± 4503.9	86.08 ±13.02	1343
OptAPO	1202.16 ± 202.07	57.23 ± 5.51	67
DPOP	228	19451.22 ± 222.26	4331

Table 6: Network 7x7: average number and size of messages

Alg.	Nb. of Msgs	Msgs size (bytes)	Total (MB)
ADOPT	358962.4 ± 1690.4	125.40 ± 20.79	43958
OptAPO	4015.66 ± 68.70	93.45 ± 8.15	366
DPOP	322	69286.78 ± 721.12	21787

Table 7: Network 9x9: average number and size of messages

As for ADOPT, the main point remains the number of cycles and exchanged messages necessary to reach a solution. It seems that, at least in this domain, ADOPT is more susceptible to the variations in the constructions of the constraints. Thus ADOPT performs differently in different time frames of a single simulation. For example, in the beginning, when there are less constraints that have high costs (hence less conflicts), ADOPT performs better than the other algorithms in terms of running time. However, as the number of constraints with high cost increases, ADOPT needs much more time to reach a solution. This cannot be seen in Table3 because it shows only the average over the whole simulation time. Another issue is that ADOPT tends to have high deviations from one simulation to the other.

In general, we noticed that the algorithm that dealt better with the increase of the number of agents was DPOP. However the issue of size of the messages remains (see Figure 4) and DPOP had its overall performance affected by this. It would have performed better than others if there were the same kind of commitment on the size of messages, as it has on the number of messages and on the number of cycles. This kind of improvement is discussed in [16].

4.5 Network Evaluation

To analyse the quality of the solution, we start by discussing Situation 1 (as defined in Section 4.2) i.e. with constant insertion rate. This situation is interesting to study because the insertion rates are the same in both NS/SN and EW/WE directions. Therefore, any control measure has difficulty coping with traffic patterns given that agents must synchronize in *one* direction only and both can be congested. For this situation, in order to assess the performance of the DCOP algorithms in different grid sizes we use mean density and mean number of stopped vehicles over the whole simulation period

Net.	Experiment	Stopped Veh.	Density
3x3	Case 1	10.21 ± 2.28	0.29 ± 0.06
	Case 2	6.72 ± 1.37	0.23 ± 0.03
	Case 3 (ADOPT)	1.88 ± 0.53	0.11 ± 0.02
	Case 3 (OptApo)	1.97 ± 0.60	0.11 ± 0.01
	Case 3 (DPOP)	2.42 ± 0.46	0.13 ± 0.02
5x5	Case 1	9.22 ± 2.36	0.28 ± 0.04
	Case 2	6.95 ± 1.82	0.23 ± 0.03
	Case 3 (ADOPT)	1.78 ± 0.80	0.11 ± 0.01
	Case 3 (OptApo)	1.90 ± 0.59	0.12 ± 0.01
	Case 3 (DPOP)	3.08 ± 0.40	0.14 ± 0.02
7x7	Case 1	8.91 ± 2.29	0.27 ± 0.04
	Case 2	5.82 ± 1.31	0.21 ± 0.02
	Case 3 (ADOPT)	1.97 ± 0.50	0.12 ± 0.01
	Case 3 (OptApo)	1.91 ± 0.49	0.12 ± 0.01
	Case 3 (DPOP)	2.90 ± 0.62	0.14 ± 0.01
9x9	Case 1	8.05 ± 2.17	0.14 ± 0.01
	Case 2	5.35 ± 0.97	0.10 ± 0.01
	Case 3 (ADOPT)	2.04 ± 0.77	0.12 ± 0.01
	Case 3 (OptAPO)	1.98 ± 0.85	0.11 ± 0.01
	Case 3 (DPOP)	2.88 ± 0.72	0.13 ± 0.02

Table 8: Comparison of performance among control via DCOP, with fixed plans, and without synchronization

(Table 8). Performances of the DCOP algorithms are compared with the cases where signal plans are not synchronized (case 1) or are synchronized in a fixed direction (case 2). After, we discuss Situation 2 (with changing insertion rates).

4.5.1 Constant Insertion Rate

In general, the use of a DCOP algorithm (case 3) has yielded an improvement regarding all metrics (defined in Section 3.6), in comparison to case 2 or case 1. Here, DPOP has performed slightly below the other two algorithms: more stopped vehicles and hence higher densities. We now discuss this in detail for all four network sizes.

Table 8 shows the average values for stopped vehicles and density. Averages are over the entire simulation time, 10 repetitions, for networks of size 3x3 (9 agents), 5x5 (25 agents), 7x7 (49 agents), and 9x9 (81 agents) respectively. It is possible to note a clear reduction on the number of stopped vehicles when DCOP algorithms are used, as well as lower density values. This positive performance can be observed in all sizes of networks.

4.5.2 Changing Insertion Rate

Here tabular results would not provide a good picture because the simulation time is divided in 6 time frames with completely different characteristics (Table 2); averages would not convey the details. Therefore we plot the curves for density along time, for all three DCOP algorithms and compare them with the cases where signal plans are not synchronized (case 1) or are synchronized in a fixed direction (case 2). We do not show all plots due to lack of space. Figure 5 shows results for the network with 49 agents. For the *Before 6 a.m.* period there is no difference between the control types due to the low insertion rate.

A different picture can be observed regarding the *Morning* and *Afternoon* scenarios, where the DCOP control – no matter the algorithm – shows that it is more flexible: when insertion rates change, all DCOP algorithms are able to keep the density lower than the other two types of control. In the *Afternoon* scenario in particular, the synchronized plans completely mismatch the insertion rate, given that they were designed to deal with higher insertion rates in the EW/WE directions, but the higher rates come actually from the NS/SN direction. In the Morning scenario, the higher insertion rate does agree with the synchronized plans.

The improvement yielded by DCOP algorithms is also noticed on the *Rush Hour I* scenario. In *Rush Hour II*, due to the high insertion rates in *both* traffic directions, which affect the synchronization results (since, as said, synchronization cannot cope with both directions), the performance is almost as good as if the plans were synchronized in a fixed way, that means in one direction. The only exception is for DPOP.

Regarding the performance of DCOP algorithms with the increase of the network, we notice that the performance depends on this size. Best results were achieved in the 5x5 and 7x7 networks where some form of control makes more sense. In the 3x3 network, there is probably no room for a sophisticated control, given the simple network; also, travel times tend to be fast because of the size of the network. Contrarily, in the 9x9 network, the volume of vehicles is higher and the size of the problems – in terms of number of constraints – is huge. Despite this, except for the Rush Hour II, DCOP was able to cope with the volume of traffic, performing at least as good as a fixed synchronization.

4.5.3 Performance in Terms of Number of Groups

The last issue we want to discuss is how big the groups formed by the DCOP algorithms are. This is important because the aim of our approach to synchronization is to emerge groups of neighbor agents running coordinated signal plans. This is not possible with

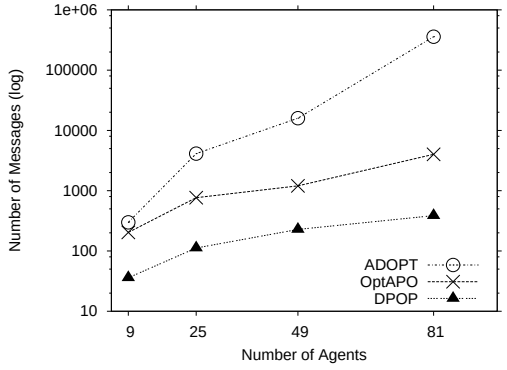

Figure 3: Number of exchanged messages (log scale) for the three algorithms, for 9, 25, 49 and 81 agents (situation 2).

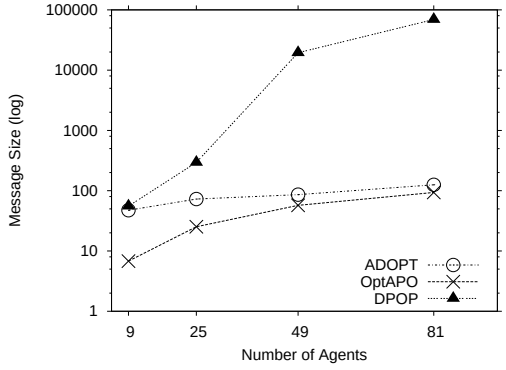

Figure 4: Size of exchanged messages (log scale) for the three algorithms, for 9, 25, 49 and 81 agents (situation 2).

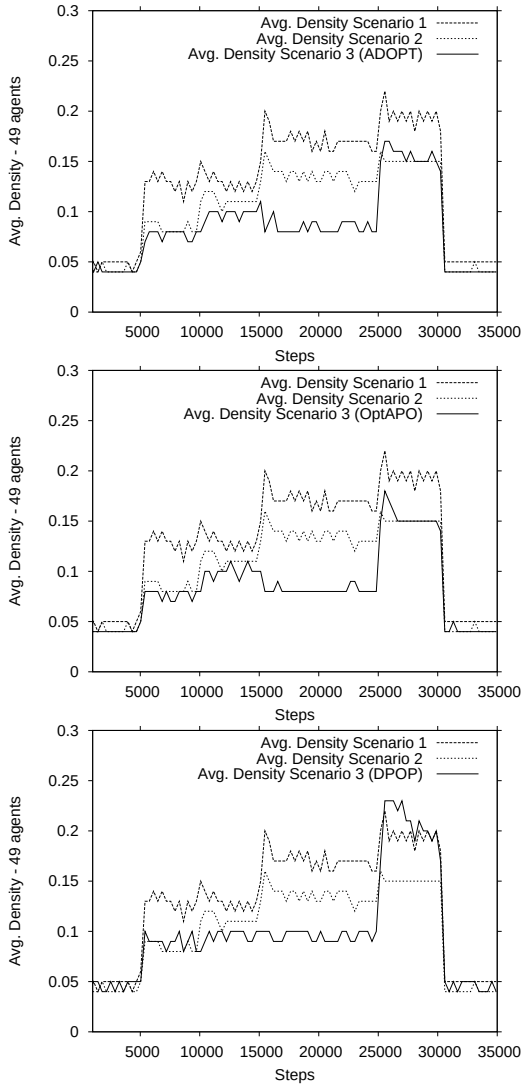

Figure 5: Average density for situation 2, network 7x7, running ADOPT (up), OptAPO (middle) and DPOP (below).

current algorithms (TRANSYT, TUC, etc.).

Thus, the analysis of the number of groups, as well as their sizes measures how agents are organizing themselves as the traffic pattern changes. One important issue is whether they are all synchronizing in accordance to the traffic direction that has more traffic volume. If not all are doing this, how many are?

We discuss here the most complicated situation regarding this issue, namely when both traffic directions get the same input of vehicles, namely a rate of insertion of 0.3. Since this rate is the same in both directions, we expect agents to be in conflict (as they can only synchronize in one of these directions). Other algorithms mentioned (e.g. TUC) resolves these conflicts manually. In our case too many conflicts usually mean that the size of the groups tend to be small.

We can consider the quality of the DCOP control (no matter the algorithm) as a good one if at least groups of a reasonable size form. Reasonable is grid size dependent. Here we consider that a group is formed when at least half of the agents in one street are synchronized. Thus, in the 5x5 grid for example, we consider a group is formed when 3 or more agents in one street are synchronized. For this grid size, there can be at most 5 groups of 5 agents each. These 5 groups are, all, synchronize either in the horizontal (EW/WE) or in the vertical (NS/SN) direction. However this configuration is difficult to reach when simulating a grid with similar insertion rates. From Table 9 one can see that even in this extreme scenario, groups of size above 3 have formed. In this table we give the average of the number of groups that form as well as their average sizes. The last column also shows the maximum size observed for each grid size.

Algorithm / Net. Size	Nb. Groups		Size Groups		Max. Size	
	hor.	vert.	hor.	vert.	hor.	vert.
ADOPT 25	3.25	3.75	3.25	3.50	4	4
OptAPO 25	3.75	3.75	3.25	3.25	4	4
DPOP 25	3.00	3.25	3.0	2.75	4	3
ADOPT 49	6.0	4.75	4.5	4.25	5	5
OptAPO 49	6.25	5.50	4.25	4.50	5	5
DPOP 49	5.25	5.00	3.25	3.50	5	4
ADOPT 81	7	7.5	5.5	5.75	7	6
OptAPO 81	7	7.25	5.75	5.25	7	6
DPOP 81	6.75	7	5.25	5.0	7	6

Table 9: Groups of synchronization: average number of groups, size of groups, maximum size of groups

5. CONCLUSION AND FUTURE WORK

The aim of this paper was to compare the performance of DCOP algorithms in a real world, dynamic scenario, in terms of computational complexity and quality of the solution. We have used the domain of traffic light synchronization, a problem of assignment of a coordinated signal plan to each two crossings in a traffic network. For this kind of problem classical solutions are all centralized [6, 7, 4] and even offline [19]. Moreover, those approaches cannot solve conflicts at network level without the intervention of rules or a human expert. We depart from centralization and model the problem as a DCOP. This is intended to be a compromise between that totally autonomous coordination with implicit communication (e.g. [1, 13]) and the classical centralized solution.

It was possible to verify the effectiveness and efficiency of the DCOP algorithms in the proposed application. Regarding the former, adaptability is a key feature in the kind of problem tackled here, where the changes in the system need to be considered and evaluated in order to guarantee an efficient control. Specifically in the traffic control problem, variations in the traffic volume cannot be totally predicted; thus they are not considered in the fixed control approaches. Changes in volume could be also the result of external factors, like accidents or meteorological conditions. Regarding efficiency, we have shown that the three algorithms perform well (e.g. within the decision-making time frame available to agents). In general the three show more or less the same performance. Of course, there are issues related to number of messages exchanged (DPOP), running time (ADOPT), and centralization (OptAPO). These tend to be critical in real world applications. Which algorithm to use will have to be determined on a case basis, considering issues such as whether there is strong limitation in communication; whether the control has to react quickly; whether the non local data can be accessed; and so on.

In general we can say that performances obtained point to the appropriateness of using DCOP in real-world scenarios with the size of those discussed here. Note that these are about the sizes some classical approaches to synchronization can deal with (e.g. TUC) but then the problem remains that conflicts must be solved

manually in the latter case.

We are already working on some extensions of the work reported here. One is new experiments with different cost functions for the DCOP and different metrics as for instance equivalent constraint checks. Another is to use the groups formed by the DCOP approach as a first step to other control approaches.

Finally, a comparison with a COP (centralized) can be performed. This is perhaps the only way to make a comparison with a centralized approach. Another obvious centralized approach could be a greedy one where each traffic light runs a signal plan that fits the global traffic state. The only problem here is that it is not trivial to define which is this state. Other approaches such as the classical ones cannot be used due to the fact that they do not work for the whole grid unless conflicts are solved by an expert; moreover those are commercial tools. Approaches based on centralized reinforcement learning have computational cost that is prohibitive for networks other than that of size 3x3 due to the number of pairs state-action.

Acknowledgments

We thank the anonymous reviewers for constructive discussions and suggestions. We are also grateful for the support of CNPq.

6. REFERENCES

[1] A. L. C. Bazzan. A distributed approach for coordination of traffic signal agents. *Autonomous Agents and Multiagent Systems*, 10(1):131–164, March 2005.

[2] M. Benisch and N. Sadeh. Examining DCSP coordination tradeoffs. In *Proceedings of the 5th International Joint Conference on AutonomousA gents and Multiagent Systems*, pages 1405–1412, New York, NY, USA, 2006. ACM.

[3] E. Camponogara and W. Kraus Jr. Distributed learning agents in urban traffic control. In F. Moura-Pires and S. Abreu, editors, *EPIA*, pages 324–335, 2003.

[4] C. Diakaki, M. Papageorgiou, and K. Aboudolas. A multivariable regulator approach to traffic-responsive network-wide signal control. *Control Engineering Practice*, 10(2):183–195, February 2002.

[5] K. Dresner and P. Stone. Multiagent traffic management: A reservation-based intersection control mechanism. In N. Jennings, C. Sierra, L. Sonenberg, and M. Tambe, editors, *The Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 530–537, New York, USA, July 2004. New York, IEEE Computer Society.

[6] P. B. Hunt, D. I. Robertson, R. D. Bretherton, and R. I. Winton. SCOOT - a traffic responsive method of coordinating signals. TRRL Lab. Report 1014, Transport and Road Research Laboratory, Berkshire, 1981.

[7] P. Lowrie. The Sydney coordinate adaptive traffic system - principles, methodology, algorithms. In *Proceedings of the International Conference on Road Traffic Signalling*, Sydney, Australia, 1982.

[8] R. T. Maheswaran, M. Tambe, E. Bowring, J. P. Pearce, and P. Varakantham. Taking DCOP to the real world: Efficient complete solutions for distributed multi-event scheduling. In *Third International Joint Conference on Autonomous Agents and Multiagent Systems*, volume 1, pages 310–317, Washington, DC, USA, July 2004. IEEE Computer Society.

[9] R. Mailler and V. Lesser. Solving distributed constraint optimization problems using cooperative mediation. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, 3.*, pages 438–445, New York, 2004. New York, IEEE Computer Society.

[10] P. J. Modi, W. Shen, M. Tambe, and M. Yokoo. ADOPT: Asynchronous distributed constraint optimization with quality guarantees. *Artificial Intelligence*, 161:149–180, January 2005.

[11] L. Nunes and E. C. Oliveira. Learning from multiple sources. In N. Jennings, C. Sierra, L. Sonenberg, and M. Tambe, editors, *Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multi Agent Systems, AAMAS*, volume 3, pages 1106–1113, New York, USA, July 2004. New York, IEEE Computer Society.

[12] D. Oliveira, A. L. C. Bazzan, and V. Lesser. Using cooperative mediation to coordinate traffic lights: a case study. In *Proceedings of the 4th International Joint Conference on Autonomous Agents and Multi Agent Systems (AAMAS)*, pages 463–470. New York, IEEE Computer Society, July 2005.

[13] D. Oliveira, P. Ferreira, A. L. C. Bazzan, and F. Klügl. A swarm-based approach for selection of signal plans in urban scenarios. In *Proceedings of Fourth International Workshop on Ant Colony Optimization and Swarm Intelligence - ANTS 2004*, volume 3172 of *Lecture Notes in Computer Science*, pages 416–417, Berlin, Germany, 2004.

[14] M. Papageorgiou, C. Diakaki, V. Dinopoulou, A. Kotsialos, and Y. Wang. Review of road traffic control strategies. *Proceedings of the IEEE*, 91(12):2043–2067, December 2003.

[15] A. Petcu and B. Faltings. A scalable method for multiagent constraint optimization. In L. P. Kaelbling and A. Saffiotti, editors, *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, pages 266–271, Edinburgh, Scotland, August 2005. Professional Book Center.

[16] A. Petcu and B. Faltings. MB-DPOP: A new memory-bounded algorithm for distributed optimization. In M. M. Veloso, editor, *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1452–1457, Hyderabad, India, January 2007.

[17] B. C. d. Silva, E. W. Basso, A. L. C. Bazzan, and P. M. Engel. Dealing with non-stationary environments using context detection. In W. W. Cohen and A. Moore, editors, *Proceedings of the 23rd International Conference on Machine Learning ICML*, pages 217–224. New York, ACM Press, June 2006.

[18] B. C. d. Silva, R. Junges, D. Oliveira, and A. L. C. Bazzan. ITSUMO: an intelligent transportation system for urban mobility. In P. Stone and G. Weiss, editors, *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006) - Demonstration Track*, pages 1471–1472. ACM Press, May 2006.

[19] TRANSYT-7F. *TRANSYT-7F User's Manual*. Transportation Research Center, University of Florida, 1988.

[20] M. Yokoo, E. H. Durfee, T. Ishida, and K. Kuwabara. The distributed constraint satisfaction problem: Formalization and algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 10(5):673–685, 1998.