SELF-MANAGEMENT IN ROAD TRAFFIC NETWORKS

Falilat Jimoh and T. L. McCluskey
University of Huddersfield, Queensgate, Huddersfield HD1 3DH, UK

ABSTRACT

Advanced urban traffic control systems are often based on feed-back algorithms. For instance, current traffic control systems often operate on the basis of adaptive green phases and flexible co-ordination in road (sub) networks based on measured traffic conditions. However, these approaches are still not very efficient during unforeseen situations such as road incidents when changes in traffic are requested in a short time interval. Therefore, we need self-managing systems that can plan and act effectively in order to restore unexpected road traffic situations into the normal order. A significant step towards this is exploiting Automated Planning techniques which can reason about unforeseen situations in the road network and come up with plans (sequences of actions) achieving a desired traffic situation. In this paper, we introduce the problem of self-management of a road traffic network as a temporal planning problem in order to effectively navigate cars throughout a road network. We demonstrate the feasibility of such a concept and discuss our preliminary evaluation in order to identify strengths and weaknesses of our approach and point to some promising directions of future research.

Keywords automated planning, urban traffic control, control systems

1 INTRODUCTION

Several types of advanced, intelligent traffic control systems operate on the basis of adaptive green phases and flexible co-ordination in (sub) networks based on measured traffic conditions [1], [2]. Where historic data is available, or where behaviour is caused by anticipated circumstances, these systems have proved very effective. For example it is possible to train signal heads to learn traffic pattern for a period of time. Such signal heads will be able to control traffic as long as the traffic pattern is closed to the learned pattern. It has been proven that learning based approaches can improve the performance of Urban Traffic Control (UTC) systems, by enabling centralized or decentralized learning and decision making within the system [3], [4], [5], [6].

These adaptive approaches to traffic control are still not optimal during unforeseen situations such as road incidents, road works, car breakdowns, and simply when traffic demand changes rapidly within a short time interval. In such circumstances, the best that are usually achieved is the use of some fixed signal timings or set of default programmed behaviour, until the situation reverts back to a recognized state [7][8]. To cope with such a situation, a system needs to be able to consider the factors affecting the situation at hand: the road network, the state of traffic flows, the road capacity limit, availability of roads within the network to mention a few. All these factors will be peculiar to the particular set of circumstances causing the problem [9]. Hence there is a need for a system that can reason with the capabilities of the control assets, and the situation's parameters as sensed by road sensors and generate a set of actions or decisions that can be taken to alleviate the situation. In this paper, we argue and demonstrate that systems that provide such dynamically created plans can be obtained from the use of automated planning technology.

The field of Artificial Intelligence Planning, or AI Planning, has evidenced a significant advancement in planning techniques in the last 10 - 15 years, which has led to development of efficient planning systems [10], [11], [13] that can input expressive models of applications. The existence of these general planning tools has motivated engineers in designing and developing complex application models which closely approximate real world problems [14], [15], [16]. Hence, AI Planning has a growing role in realisation of intelligent control systems. The long term goal of this research field is to exploit a traditional control system architecture, which is based on reactive reasoning, and embed it with situational awareness, together with declarative representation of goals, actions and states of its environment, and explore the possibility of using planning engines to support deliberative reasoning within this system.

In this paper, we explore the application of AI planning technology to the problem of Urban Traffic Control (UTC) system. We investigate the feasibility of using this approach in an urban road traffic domain to generate plans dynamically, where learnt or reactive behaviours fail because of unforeseen situations. We show how this is accomplished by the creation of a declarative representation of a road network in a town center area in the United Kingdom. The task is to effectively navigate cars through network of connected roads during an unforeseen situation. As part of this effort, we have embed the knowledge of UTC structure into a planning domain and evaluate the possibility of reasoning with this knowledge and optimising traffic flow in situations where a given road within a network of roads becomes unavailable due to unexpected situation such as road accidents. From this simulation we demonstrate the advantages and shortcomings of using AI Planning in UTC, i.e., navigating cars though the road network and eventually diverting cars in case that some unplanned circumstance causes a road blockage.

3 THE ROLE OF AUTOMATED PLANNING IN CONTROL SYSTEMS

Automated Planning also know has AI planning, deals with the problem of finding a sequence or partially ordered set of actions whose execution leads from an initial state to a state in which a goal condition is satisfied. Actions in plans are ordered in such a way that executability of every action is guaranteed. Hence, an agent is able to transform the environment from an initial state into a desired goal state. A planning problem thus involves deciding "what" actions to do, and "when" to do them.
Technology in areas such as automated planning and constraints processing have been developing rapidly, so nowadays it is possible to deploy deliberative reasoning to real-time control applications [22], [23], [24], [25]. The main difference between traditional and deliberative control architecture is depicted in Figure 1. Traditionally, a control loop consists of three steps: sense, interpret and act [26]. In other words, data are gathered from the environment with the use of sensors, the system interprets information from these sensors as the state of the environment. The system acts by taking necessary actions, these actions are feedback into the system in other to keep the environment in desirable state. Introducing deliberation in the control loop allows the system to reason and generate effective plans in order to achieve desirable goals. Generally speaking, using deliberative reasoning enables achieving more 'distant' goals rather than immediate ones. Enabling deliberative reasoning in UTC systems is important because of its ability to handle unforeseen situations which has not been previously learnt nor programmed into a UTC. This helps to reduce traffic congestion and carbon emissions on our roads. Thus, embedding automated planning into UTC systems to enables deliberative reasoning in order for urban traffic controllers to generate plans and schedules to manage themselves would be very beneficial in the realisation of self-management in such systems.

4 URBAN TRAFFIC CONTROL PROBLEM

Current traffic control systems approaches are still not completely optimal during unforeseen situations such as road incident, road re-construction, car breakdown and when traffic demand changes rapidly within a short time interval. In this situation, there arises the need for systems that can take pro-active decision using deliberative reasoning to control the road traffic situation in order to achieve desirable goals. Ideally, the following abilities would be incorporated into a road traffic control system.

- Ability to retrieve information on the current state of traffic based on information from road sensors
- Ability to detect current traffic problems by evaluating current traffic trends with the use of traffic control rules on an existing ideal model of the surrounds
- Ability to estimate the next optimal cycle of signal heads
- Ability to take decision and operate the entire traffic control signals based on it generated plan with little or no human intervention
- Ability to communicate with road users in real time.

4.1 Resources and Constraints

In every system design, there are resources available for execution. These resources come with associated limitations (constraints) which must be identified and optimised for effective implementation of the system. In UTC we are dealing with various resources often limited by constraints. For example, we have road junctions, which can be understood as bottlenecks of traffic, where we must ensure that cars are not permitted to go through the junction in colliding directions.

Also, each road had its capacity, i.e., the maximum number of cars on it. Situational awareness of the system can be sustained by various sensors which are placed on roads, intersections and so on.
Each action has its own duration in which is being executed. For instance, the duration of action for vehicle movement from a head to a tail of the road depends on the road's length and the speed of the vehicle.

5 THE ROAD TRAFFIC DOMAIN MODEL

In order to explore the role of automated planning in enabling some of the requirements described in the sections above, this section describes the design of a Road Traffic Domain Model (RTDM). From the automated planning perspective we have to compromise between making RTDM realistic and the computational resources for solving RTDM planning problems. A very realistic RTDM should be able to reason about continuous processes (e.g. cars are moving on the road continuously), uncertainty (e.g. a driver can decide its own way) and unexpected events (e.g. traffic accidents). Continuous planning without reasoning about uncertainty is not well developed. On the other hand, classical planning (the simplest form of planning) is very well developed; however, not very suitable for RTDM since classical planning does not reason about time (i.e. effects of actions are instantaneous). Hence, the reasonable compromise is using temporal planning which can in addition to classical planning reason about time (i.e. executing actions takes some time).

RTDM consists of two main parts – static and dynamic. The static part represents road network topology, i.e., roads, their capacity, length and junctions connecting the roads. The dynamic part stands for how many cars are on each road (and where) and whether the road is operational. The term 'operational' means that the road is available and accessible within the road network system. Clearly, the dynamic part is changing through the time as cars are moving through the road network.

5.1 RTDM Specification

A Road Network can be represented by a directed graph, where edges stand for roads and vertices stand for either junctions, entry or exit points. Entry points are places where cars enter the network, while exit points are places where cars exit the network. In junctions, we must take into account the fact that in some directions cars cannot go through the junction simultaneously. Hence, we have to specify sets of mutually exclusive (mutex) directions by which cars cannot go through the junction simultaneously. Every road has its own length (determining how long it takes for a car to through it) and capacity (i.e. a maximum number of cars it can serve). The network model is enhanced by considering time intervals when a road is not operational (e.g. closed due to an accident).

A RTDM Planning Problem addresses the problem of effective navigation of cars through a given Road Network from entry points to exit points. To achieve this we have to take two types of decisions, i.e., which way cars should take and when traffic lights should be set to green or red. Initially, it is given the number of cars in each entry point and frequency of their releasing. This can be represented by a set time-stamps in which the entry points are 'opened'. The goal situation is determined by numbers of cars in exit points while minimising 'makespan', the time needed to navigate all the cars through the road network. In our model we consider four planning operators.

5.2 Modelling RTDM in PDDL

Planning Domain Description Language(PDDL)[27] is currently a standard language for describing planning domains and problems and it is widely accepted by state-of-the-art planning engines.
For our purpose we have use PDDL 2.1 [17] since it encapsulates features needed for representing temporal planning domains and problems. The environment of RTDM is modelled by using predicates or fluents. Predicates refers to relations between objects, for example, if a predicate (*operational r1*) is present in some state, then in this state the road *r1* is operational, otherwise not. Fluents can operate with more (numerical) values, for instance, a fluent (*use r1*) represents a current use of the road *r1* which is in range *<0,C(r1)>.*
Figure 3 depicts the planning operator *drive* (in PDDL 2.1 planning operators are called 'durative actions'). Time-stamps in which the operator can be executed are not explicitly modelled as operator's parameter in PDDL, only duration of operator's execution is defined *:duration (= ?duration (length ?r)).* Time-stamps are, therefore, modelled relatively. The operator is executed in a time-stamp *t* and the duration of its execution is Δt, at start refers to *t*, *at end* refers to *t* + Δ and *over all* refers to the interval <*t, t + Δt*>. To keep the consistency of the environment we cannot apply effects at the same time the precondition is being checked (e.g. apply *(at start (decrease (head ?r) (val ?n)))* at the same time

when *(at start (>= (head ?r) (val ?n)))* is being checked). This issue is handled by applying the effect just after the precondition has been checked, so for a very small ϵ the effects are applied in t + ϵ *(at start)* or in t + Δt + ϵ *(at end).* Another issue in PDDL 2.1 is impossibility to have a numeric parameter. This issue can be handled by introducing a special object type num and a fluent *(val ?n)* which maps a numerical values to num objects. In all considered models, the goal of the planning problem is to route a certain number of vehicles through the network to the exit points, whilst minimising the makespan of the plan irrespective of any disruption in the network of connected roads.

A typical RTDM problem requires a method of releasing cars at the entry points at given (periodic) time-stamps. Releasing cars can be understood as an event, however, events are not supported in PDDL 2.1. Releasing cars by the operator release-cars is executed in any time when its precondition is met. This issue can be overcome by using timed initial literals which is a feature supported by PDDL 2.1. We introduce a fluent (ready ?x) for each entry point, representing the number of cars which are ready to enter the road network.

Therefore, several statements of the form:

```
(at 0 (= (ready a) 5))
(at 10 (= (ready a) 5))
          ....
```

Is added to the initial state. We believe that this is an attractive approach as it ensures the flow of traffic into the system in a more realistic way rather than simply adding all vehicles at time 0. We also use timed initial literals for defining temporary road blockages.

6 PRELIMINARY EVALUATION

For evaluation purposes we selected known planning engines Crikey [11], Optic [28] and LPG-td [10] which are capable of handling PDDL 2.1 features (including timed initial literals). LPG-td, however, is not very useful for our problem by two reasons. Firstly, it does not successfully complete pre-processing when the problem has more than a few objects (e.g. roads). Secondly, it cannot handle well concurrency from actions sharing some fluent(s), i.e., actions having the same fluent in their descriptions are never considered as concurrent.

The experimental setting consists of several ‘specific’ problems which are defined on the top of the road network depicted in Figure 2. Uncertainty is not considered in our experiments. Problem 1 addresses the problem of navigating five cars from Lord Street (bottom right corner of the map) to Wood Street (upper part of the map). Problem 2 addresses the problem navigating cars (two at each entry point) through the network such that they are evenly distributed at south, north and east exit points (individual vehicles are not distinguished). Problem 3 has the same settings as but has five cars at each entry point. Problems 1-3 are considered in two different variants, i.e., without blockages and with blockages. Experiments were run on Intel Core i7 2.9GHz, 5GB RAM, Ubuntu 12.04.1 LTS.

Table I. shows the results of our preliminary experimental settings using the Crikey and Optic planners. We can see that Optic clearly outperforms Crikey in both criteria – makespan (a time needed for executing the plan) and runtime (a time needed by a planner to provide a plan). Moreover, Optic is an incremental planner, i.e., after finding a plan it searches for a better one (with lower makespan) until a given time limit is reached. The results of Optic are very promising given the fact that plans have been retrieved in a very reasonable time (at worst slightly above one second) and their quality (makespan) is satisfactory. Cars can be therefore navigated reasonably through the road network even in case of some unexpected event which could lead to road blockage. Also, given the fact that in the real-world environment cars are entering the road network contiguously, which we have not been able to reasonably model (it is discussed later), good quality of plans (in terms of makespan) can somehow ensure that the traffic flow remains continuous and roads will not become congested.

The generated plans (not shown here because of lack of space) show some interesting aspects. Firstly, despite going from the same entry point to the same exit point cars are split during the way such that some cars are navigated through Northumberland Street

(e1) and some cars are navigated through St. Peter Street (v2). This might be useful when some unexpected event occurs (e.g. an accident on St. Peter Street) and there is no time to redirect traffic. Secondly, the Optic planner is not much able to consider more than one car in a single action (nn1) even though it is possible. This is a shortcoming because it does not lead to optimal (or very nearly optimal) solutions. Basically, such a plan when executed allows one car to go through the junction, this means the other cars have to wait until that car drives through the following road. This is quite counter-intuitive and consequently to this plans are not optimal. Another observation is that both Crikey and Optic do not handle timed-initial literals well. For example, if a road is blocked only for a given time interval, then the planners conclude that there does not exist a solution despite the fact that a solution exist for the same problem whenever it is blocked for entire planning duration.
In summary, embedding planning component into the UTC might be beneficial since it enables (centralised) reasoning in the given area which, for instance, can more easily overcome non-standard situation (e.g. road blockage after an accident).

Our results showed that the makespan of the plans (given by the Optic planner) did not increase much even though some roads were blocked. Minimising makespan, which is a goal of any UTC system, will help to reduce road congestion and pollution in the environment. However, as we demonstrated using state-of-the-art domain-independent planning engines does not lead to optimal solutions even in quite simple cases. Also, it is questionable whether planners' performance will not significantly decrease on larger road networks. On the other hand, developing a domain-dependent planner specifically tailored to RTDM might overcome (some of) these issues.

7 CONCLUSION

In this paper, we addressed the idea of introducing Automated Planning into UTC which, moreover, can reroute traffic flow when a road becomes unavailable due to unexpected circumstances. As part of this effort, we have embedded the knowledge of UTC structure into a planning domain and evaluated the possibility of reasoning with this knowledge and optimising traffic flow in situations where a given road within a network of roads becomes unavailable due to unexpected situation such as road accidents. This allows us to navigate cars throughout the road network. Our preliminary experimental evaluation showed that our approach is able to provide plans in a reasonable time. In a real-world scenario where data such as road queues are uploaded in real-time from road sensors with traffic signals connected to a planner, we believe that our approach can divert road traffic from a blocked road without human intervention.

In future, we plan to embed our model into a road traffic simulation environment. We also plan to provide a deeper evaluation of our approach, especially to compare it with traditional traffic control methods, and assess the effort and challenges required to embody the model within a real-world environment. Other aspects of improvement would include: incorporating the actual road policies; increase interaction with road users; build a knowledge base from generated (optimal) plans; consider various speed limits; consider weather conditions and priority vehicles.

8 REFERENCES

[1] D. A. Roozemond, "Using intelligent agents for pro-active, realtime urban intersection control," European Journal of Operational Research, vol. 131, no. 2, pp. 293–301, 2001.
[2] D. De Oliveira and A. L. C. Bazzan, Multiagent Learning on Traffic Lights Control: Effects of Using Shared Information, 2009, pp. 307–322.
[3] Z. sheng Yang, X. Chen, Y. shan Tang, and J.-P. Sun, "Intelligent cooperation control of urban traffic networks," in Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on, vol. 3, 2005, pp. 1482–1486 Vol. 3.
[4] A. Salkham, R. Cunningham, A. Garg, and V. Cahill, "A collaborative reinforcement learning approach to urban traffic control optimization," in Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 02, ser. WI-IAT '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 560–566. [Online]. Available:
http://dx.doi.org/10.1109/WIIAT.2008.88
[5] F. Daneshfar, J. RavanJamjah, F. Mansoori, H. Bevrani, and B. Z. Azami, "Adaptive fuzzy urban traffic flow control using a cooperative multi-agent system based on two stage fuzzy clustering," in Vehicular Technology Conference, 2009. VTC Spring 2009. IEEE 69th, 2009,
pp. 1–5.

[6] A. L. Bazzan, “A distributed approach for coordination of traffic signal agents,” Autonomous Agents and Multi-Agent Systems, vol. 10, no. 1, pp. 131–164, Jan. 2005. [Online]. Available: http://dx.doi.org/10.1007/s10458-004-6975-9
[7] I. Dusparic and V. Cahill, “Autonomic multi-policy optimization in pervasive systems: Overview and evaluation,” ACM Trans. Auton. Adapt. Syst., vol. 7, no. 1, pp. 11:1–11:25, May 2012. [Online]. Available: http://doi.acm.org/10.1145/2168260.2168271
[8] X.-F. Xie, S. Smith, and G. Barlow, “Schedule-driven coordination for real-time traffic network control,” in International Conference on Automated Planning and Scheduling, 2012. [Online]. Available: http://aaai.org/ocs/index.php/ICAPS/ICAPS12/paper/view/4701/4744
[9] F. Jimoh, L. Chrpa, P. Gregory, and T. McCluskey, “Enabling autonomic properties in road transport system,” in The 30th Workshop of the UK Planning And Scheduling Special Interest Group, PlanSIG 2012, 2012.
[10] A. Gerevini, A. Saetti, and I. Serina, “An approach to temporal planning and scheduling in domains with predictable exogenous events,” J. of AI Research, vol. 25, pp. 187–231, 2006.
[11] A. I. Coles, M. Fox, D. Long, and A. J. Smith, “Planning with problems requiring temporal coordination,” in Proc. 23rd AAAI Conf. on Artificial Intelligence, July 2008.
[13] P. Eyerich, R. Mattm¨uller, and G. R¨oger, “Using the Context-enhanced Additive Heuristic for Temporal and Numeric Planning,” in Proc. 19th Int. Conf. on Automated Planning and Scheduling (ICAPS), 2009.
[14] J. Hoffmann and S. Edelkamp, “The deterministic part of ipc-4: An overview,” J. Art. Int. Res. (JAIR), vol. 24,pp. 519–579, 2005.
[15] M. Fox and D. Long, “Modelling mixed discrete-continuous domains for planning,” J. Art. Int. Res. (JAIR), vol. 27, pp. 235–297, 2006.
[16] P. Haslum and H. Geffner, “Heuristic planning with time and resources,” in Proc. 6th European Conference on Planning (ECP’01), 2001, pp. 121–132.
[17] M. Fox and D. Long, “PDDL2.1: An extension of PDDL for expressing temporal planning domains,” J. Art. Int.Res. (JAIR), vol. 20, pp. 61–124, 2003. [Online]. Available: http://www.jair.org
[22] J. Lhr, P. Eyerich, T. Keller, and B. Nebel, “A planning based framework for controlling hybrid systems,” 2012. [Online]. Available: http://aaai.org/ocs/index.php/ICAPS/ICAPS12/paper/view/4708/4726
[23] D. F. Ferber, “On modeling the tactical planning of oil pipeline networks.” in ICAPS, L. McCluskey, B. Williams, J. R. Silva, and B. Bonet, Eds. AAAI, 2012.
[24] E. Burns, J. Benton, W. Ruml, S. Yoon, and M. Do, “Anticipatory on-line planning,” in International Conference on Automated Planning and Scheduling, 2012. [Online]. Available: http://aaai.org/ocs/index.php/ICAPS/ICAPS12/paper/view/4696/4745
[25] S. Kiesel, E. Burns, C. Wilt, and W. Ruml, “Integrating vehicle routing and motion planning,” 2012. [Online]. Available: http://aaai.org/ocs/index.php/ICAPS/ICAPS12/paper/view/4709/4723
[26] M. Gopal, Control systems: principles and design. London: McGraw- Hill, 2008.
[27] McDermott D. et al., “PDDL–the planning domain definition language,” Available at: www.cs.yale.edu/homes

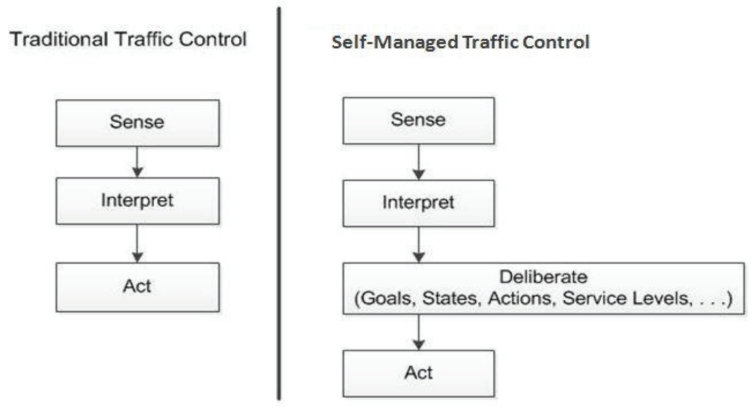
Traditional Traffic Control
Sense
Interpret
Act
Self-Managed Traffic Control
Sense
Interpret
Deliberate
(Goals, States, Actions, Service Levels, . . .)
Act

Fig. 1. Comparison of traditional and deliberative controls in Urban Transport Systems.

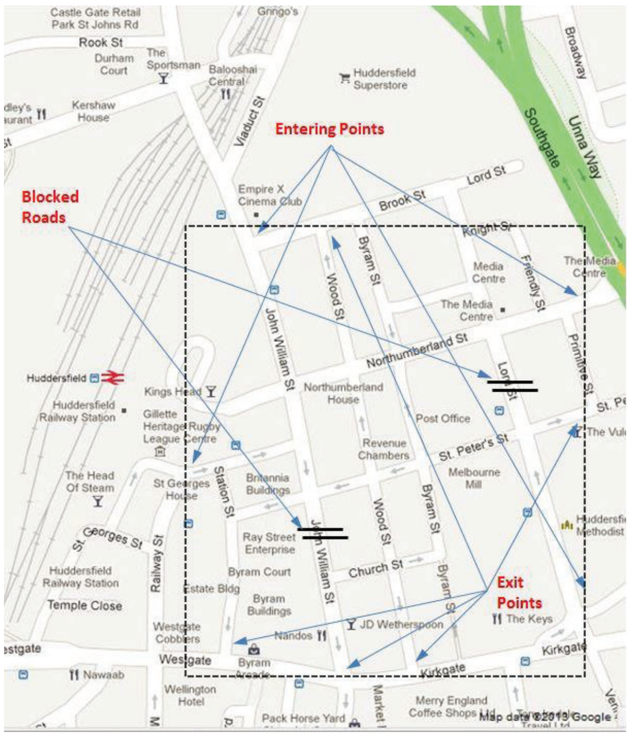

Fig. 2. Map showing the network entry and exit points and the blocked roads in a part of town center of Hudderfield, West Yorkshire, United Kingdom. It is used for our empirical analysis.

```
(:durative-action DRIVE
:parameters (?r - road ?n - num)
:duration (= ?duration (length ?r))
:condition
 (and
   (at start (>= (head ?r) (val ?n)))
   (over all (operational ?r)
 )
:effect
 (and
   (at start (decrease (head ?r) (val ?n)))
   (at end (increase (tail ?r) (val ?n)))
 )
)
```

Fig. 3. The PDDL encodings of the drive planning operator

Prob.no.	Blockages	Crikey		Optic	
		Runtime	Makespan	Runtime	Makespan
1	No	0.16	156	0.19	53
1	Yes	0.14	156	0.16	64
2	No	1.22	122	0.35	43
2	Yes	1.28	170	0.40	52
3	No	36.84	299	1.79	57
3	Yes	51.52	446	1.54	70

Table 1. Our experimental results showing planners' performance. Runtime (in seconds) stands for a time needed by a planner to produce a plan. Makespan stands for a Duration in which the plan can be executed.