Dynamic Mechanism Design for Efficient Planning under Uncertainty

TRAIL Research School, Delft, October 2012

Authors

Ir. Joris Scharpff, Dr. Mathijs de Weerdt, Dr. Matthijs Spaan

Faculty of Electrical Engineering, Mathematics and Computer Science, Department of Algorithmics, Delft University of Technology, The Netherlands

Contents

Abstract

We study the planning of maintenance activities on public infrastructural networks – road networks, Internet, power grids, etc. – in contingent environments such that the negative impact on the network user is minimised. Traditional efforts hereto are mainly of a regulatory nature, whereas we propose charging the service-providers (agents) proportional to the harm they cause, thus representing the road user implicitly. Additionally, we seek to exploit the additional opportunities of implicit coordination between agents that arise as a consequence of user cost charging.

In this paper we discuss several existing methods for efficient maintenance planning in contingent environments with interdependent agents and we propose a first attempt at a general dynamic mechanism that is to be refined in future work. By experimental analysis we show the validity of our mechanism.

Keywords

Dynamic mechanism design, Planning and scheduling

1 Introduction

Recent advances in multi-agent planning bring real-life applications within reach (e.g. Moonen (2009)). In addition, the field of algorithmic mechanism design has brought computationally feasible methods to make decisions when agents are self-interested Narahari et al. (2009). However, in real-life planning problems, such decisions need to be made in a dynamic context and with an uncertain future. For such a complex strategic setting only very few known theoretical results are applicable Athey & Segal (2007); Bergemann & Valimaki (2006); Cavallo (2008), and we are not aware of any application of these results beyond toy problems (except perhaps Seuken et al. (2008)). The contribution of this paper is to demonstrate and validate the use of a combination of algorithms for planning with uncertainty Dean et al. (1993) and mechanisms for dynamic settings to deal with the problem of coordinating multi-agent planning with interdependent agents in a contingent environment.

This problem is inspired by the real-world setting of maintenance of road infrastructure, such as a national network of highways. Often, several contractors will be responsible for maintaining different segments of the network. Large maintenance activities can impact the throughput of a highway segment: for instance, when renewing the top layer of the road, certain lanes will need to be closed. One of the goals of a national road authority is to ensure that maintenance is planned in a such that the overall throughput of the network is maximized. The crucial element here is that even though contractors are independent firms and maintain different parts of the infrastructure, the maintenance activities (lane closures) of one contractor affect other contractors as traffic flows will change throughout the network.

In the road maintenance problem a national road authority faces a mechanism design problem: how to draw up contracts that incentivise contractors to coordinate their planned maintenance activities so as to minimize the disruption of the overall traffic flow. We model the planning problem of each contractor as a Markov Decision Process (MDP) Dean et al. (1993), as it naturally captures the uncertainties that can occur when planning such activities. This paper explores several ways how these individual problems can be combined and solved, using the theory of dynamic mechanism design.

Below we start by discussing the problem of planning under uncertainty for multiple agents in a bit more detail (Section 2) and outline several solution approaches. We then focus on settings where the agents are self-interested. In Section 3 we discuss the dynamic mechanism design concepts relevant to our work, and present our mechanism using a case study on infrastructural networks in Section 4. Finally, we experimentally verify the validity of our mechanism in Section 5.

2 Efficient Planning under Uncertainty

Planning in an ideal world, where all tasks are guaranteed to succeed, is deterministic and hence we are able to develop optimal plans offline. However, in real world applications we are dealing with a contingent environment and tasks might

fail due to external factors. This might cause offline optimal plans to result in rather poor results when executed unchanged.

In order to guarantee efficiency throughout the entire execution period, we cannot rely on a single joint plan as it is not robust against contingencies. Instead, we are looking for efficient policies that dictate the action(s) to take given the *current execution state*, whether it was the state we expected beforehand or another state we end up in due to some unforeseen event.

The presence of uncertainty in planning can be captured rather naturally in the Markov Decision Process (MDP) frameworkDean et al. (1993). At each point in time the network is in a certain state, for which the agents need to decide what actions to take. The new state of the network depends on the current state and some probabilistic transition function that represents the uncertainty in the executions. Note that this does require knowledge about the possible uncertainties that can occur; performing risk-assessment in advance helps identifying such risks.

In this section we present several algorithmic approaches for finding efficient policies for planning under uncertainty and we argue that a mechanism design approach is preferable for all participating parties.

2.1 Centralised Planning

If we assume that the agent activity sets and cost functions are common knowledge (or agents are willing to disclose this information to a center), we are able to devise a centralised algorithm for developing efficient policies. We can combine all the individual agent MDPs and solve the joint MDP using a state-of-the-art MDP solver (e.g. Spudd Hoey et al. (1999)) to obtain the jointly-efficient policy.

Although this approach is most efficient in terms of total reward, it has several drawbacks. First of all, the assumption that agents are willing to disclose their private information in a competitive setting is rather strong. Secondly, this approach requires the agents to accept plans dictated by a center that might be optimal from a joint perspective but (a lot) more expensive for the agent himself.

A third practical issue is that finding efficient policies quickly becomes rather impractical from a computational point of view. We underline that this heavily depends on the encoding of the problem; in our experiments with the Spudd solver, the run time quickly becomes prohibitively large.

2.2 Individual Planning

The opposite of centralised planning is individual planning, in which agents develop their policies locally. This approach does not suffer from the autonomy and privacy issues as much (although the planner can derive information about the agent from its policy) and is computationally rather easy to solve. The drawback is of course that dependencies between agents and the effect of the policies on the user cost are completely ignored. Especially when charging the agents the part of the user cost they introduce, individual planning performs rather bad in terms of agent profit.

2.3 Best-Response Planning

Best-response planning (BRP, Jonsson Jonsson & Rovatsos (2011)) can be considered a compromise between centralised and individual planning. Agents develop an initial joint plan, e.g. individually, and iteratively improve on this plan in a Nash-like best-response way. In this way, the privacy and autonomy issues are minimised. In addition, computing the best response given a joint plan is far less computationally complex while we are still able to account for the impact of the plan on other agents and the network user (by incorporating this in the cost of a move). Note however that the BRP method depends on a given joint plan and can therefore not be easily used to find policies. Instead, we have to run the BRP algorithm in each time step, taking several iterations in each step to reach 'reasonable' solutions.

Another weakness of the BRP method is that, as we are using Nash dynamics to find joint plans, we have to settle for Nash equilibria in our solutions. This solution concept has not been widely studied for planning under uncertainty, hence we are unsure about the quality or even existence of such equilibria. In addition, the equilibria that can be reached by the BRP algorithm depends greatly on the order of moves. The BRP method can get trapped in local optima, preventing further iterations to reach (near-)optimal solutions.

2.4 Mechanism Design

As we are dealing with selfish, autonomous and rational agents, a more natural approach to solving our problem is to model the problem using *Algorithmic Game Theory (AGT)*.[1] Instead of a centralised solution in which all competitive agents need to reveal their private information, we design an incentive mechanism that aligns the individual objectives with the global goal of finding efficient joint plans. In this way it is in the best interest of players to plan efficiently, assuming they are rational.

This method overcomes the privacy and autonomy issues while still being able to produce optimal joint plans. Note that the computational aspect depends on the outcome rule and its corresponding algorithm of the mechanism that is being employed.

Nonetheless we cannot directly apply traditional mechanism design on planning problems with uncertainty for the reason stated in the introduction of this section. A 'static' mechanism is only able to produce a single plan and is therefore not robust against contingencies. Instead we focus on mechanisms in a dynamic setting. In the next section we present the theoretical background of *dynamic mechanism design*, required for the understanding of the novel dynamic planning mechanism we propose in Section 4 .

[1] For an introduction or refresher on AGT we refer the reader to for instance Nisan (2007). A short but excellent summary of the most relevant notions in the light of our research can be found in e.g. Dash et al. (2003), although there the term computational mechanism design is used.

3 Dynamic Mechanism Design

Dynamic mechanisms Athey & Segal (2007); Bergemann & Valimaki (2006); Cavallo (2008) extend 'static' mechanisms to deal with sequential games in which the private information of players evolves over time. In each time step, players[2] need to determine the best action (in expectation) to take while considering current private information and possible future outcomes. Many of the concepts defined in mechanism design have dynamic equivalents, which we present in this section. Most of the definitions in here originate from the work by Cavallo in Cavallo (2008).

Definition 3.1: Direct Dynamic Mechanism

A dynamic mechanism M is a tuple $\langle \pi, \mathcal{T} \rangle$ in which $\pi : \Theta \rightarrow \mathcal{A}$ is a decision policy that maps the joint type space to the set of all possible actions[3] *$\mathcal{A}$ and $\mathcal{T} = \{\mathcal{T}_i \mid i \in N\}$ is a set of payment functions $\mathcal{T}_i : \Theta \rightarrow \mathbb{R}$.*

An important notion in mechanism design is the *revelation principle*Gibbard (1973). This theorem states that any truthful indirect mechanism, in which agents only report outcomes. can be transformed into an incentive compatible direct mechanism, where agents disclose their full private information, and hence it suffices to study only the latter type of mechanism. An extension of the revelation principle for dynamic mechanisms also exists, due to the work by Pavan, Segal and Toikka Pavan et al. (2008).

The payment functions $\mathcal{T}_i$ defined by the mechanism map a single joint type to a payment. In addition we define the expected payment functions $\mathcal{T}_i^*$ that represent the expected payments from some joint type θ^t until the end of the finite, discrete planning period T, given the mechanism policy π and the strategies σ (from the set of possible strategies Σ) played by all the players. This function is given by $\mathcal{T}_i^*(\theta_i^t, \pi, \sigma) = \mathbb{E}\big[\sum_{k=t}^{T} \gamma^{k-t} \mathcal{T}_i(\sigma(\theta^t))\big]$ given some discount factor $\gamma \in [0, 1]$.

The strongest solution concept in 'static' mechanism design, the dominant strategy equilibrium, is more difficult to realise in dynamic mechanisms, due to the contingent future. Instead we focus on equilibrium strategies that, when all agents adhere to them, in expectation yield maximum utility for every possible joint type $\Theta = \theta_1 \times \ldots \times \theta_N$. This is defined as the within-period ex post Nash equilibrium:

Definition 3.2: Within-period ex post Nash equilibrium

A strategy σ for a dynamic mechanism M constitutes a within-period ex post Nash equilibrium if and only if at all times $t \in T$ for all players $i \in N$, all true types $\theta^t \in \Theta$ and all other strategies $\sigma' \in \Sigma$: $V_i(\theta^t, \pi, (\sigma_i, \sigma_{-i})) + \mathcal{T}_i^*(\theta^t, \pi, (\sigma_i, \sigma_{-i})) \geq V_i(\theta^t, \pi, (\sigma'_i, \sigma_{-i})) + \mathcal{T}_i^*(\theta^t, \pi, (\sigma'_i, \sigma_{-i}))$. Here V_i denotes the (expected) value player i has for executing strategy σ_i given the current joint type θ^t and the outcome chosen by π when others perform strategies σ_{-i}.

[2] We use the terms agent and player interchangeably. The term players is commonly used in game theory literature.

[3] The definition of this set depends on the encoding used. For instance, in a naive encoding for the centralised planning MDP this could be the Cartesian product of all agent activity sets.

Continuing along these lines, the notion of incentive compatibility can be extended to within-period ex post Nash equilibria. A mechanism is incentive compatible in this solution concept if for every joint type truthfully reporting maximises utility when other agents are also truthful. Mechanisms are individually rational in this equilibrium if for all joint types each agent is expected to make non-negative profit, given that all agents play a within-period ex post Nash equilibrium strategy. Finally, the mechanism is no-deficit if the sum of its payments to agents over the entire period T is non-positive. For the formal definitions of these concepts see Cavallo (2008).[4]

In Bergemann & Valimaki (2006), Bergemann and Valimaki proposed a dynamic version of the well-known VCG mechanism for the within-period ex post Nash equilibrium solution concept. Cavallo adapted this definition to fit in the dynamic mechanism framework in Cavallo (2008). The dynamic-VCG mechanism (definition 3.3) yields maximum revenue among all mechanisms that satisfy efficiency, incentive compatibility and individual rationality in within-period ex post Nash equilibrium. Therefore we only restrict our focus to this mechanism in the rest of the paper.

Definition 3.3: Dynamic-VCG

A direct, dynamic mechanism $M = \langle \pi, \mathcal{T} \rangle$ belongs to the class of dynamic-VCG mechanisms if is satisfies:

(i) The decision policy π is efficient for all time steps t, or $\forall \theta^t \in \Theta$: $\pi = \arg\max_{\pi \in \Pi} \sum_{i \in N} V_i(\theta^t, \pi)$.

(ii) In every time step t players pay related to the additional cost they incur on other players, or $\forall \theta^t \in \Theta, i \in N, \sigma_i \in \Sigma$: $\mathcal{T}_i^*(\theta^t, \pi, \sigma_i) = V_{-i}(\theta^t, \pi, \sigma_i) - \mathcal{C}_i(\theta^t, \pi, \sigma_i)$ in which $C_i : \Theta \to \mathbb{R}$ is an arbitrary function and $\mathcal{C}_i(\theta^t, \sigma_i) = \mathbb{E}[\sum_{k=t}^{T} \gamma^{k-t} C_i((\sigma_i(\theta_i^k), \theta_{-i}^k))]$, i.e. the expected (discounted) value of the function C_i.

(iii) Each player i pays exactly the expected valuation that other agents could have obtained forward from the joint type that follows from executing π without i participating, or $\forall \theta^t \in \Theta, i \in N, \sigma_i \in \Sigma$: $\mathcal{T}_i^*(\theta^t, \pi, \sigma_i) = V_{-i}(\theta^t, \pi, \sigma_i) - V_{-i}(\theta_{-i}^t, \pi, \sigma_i)$

Note that the first two conditions of definition 3.3 ensure that the mechanism is a member of the dynamic-Groves class Cavallo (2008). The third condition is the dynamic version of the Clarke tax, similar to the traditional VCG mechanism.

4 Case Study: Infrastructural Networks

In this section we propose our dynamic mechanism for planning under uncertainty, using maintenance planning for infrastructural networks as a case study to aid the

[4] A weaker solution concept, the Bayes-Nash equilibrium is also defined for dynamic mechanisms, however we will not discuss this here as we are only interested in within-period ex post Nash equilibria. We refer the interested reader to Athey & Segal (2007) or Cavallo (2008).

reader in understanding our method. This case study is motivated by the larger research project 'Dynamic Contacting in Infrastructures' in the context of which this work is being performed.

The example in this section is rather concrete, while our method is applicable on a large scale of planning problems. In general, our mechanism can be applied on any planning problem for which we want to find *efficient* plans in contingent environments where the participating agents are willing to disclose their private information. By using an indirect mechanism, this requirement can be relaxed but this is not in the scope of this paper.

4.1 Infrastructural Networks

The problem we study in this section arises in maintenance of public infrastructures such as e.g. road networks, power grids or the internet. Commonly there is a public asset manager (i.e. governmental institution, public organisation, etc.) responsible for the network quality at the behalf of the (tax-)paying user. Moreover, there is a cost associated with the use of the network, for instance travel time in road networks or packet travel times.

An infrastructural network is defined by $\mathcal{N} = \langle G, \ell, Q \rangle$. $G = \langle V, E \rangle$ is a representation of the network topology. With each edge $e \in E$ we associate a quality level q_e.[5] The cost to the user for a flow (e.g. traffic, packets, etc.) f on the network is captured by the function $\ell(f) \in \mathbb{R}$. In addition, we define the *marginal user cost* of any flow f in respect to the flow f^0 that occurs when all edges are completely available as $\ell^*(f) = \ell(f) - \ell(f^0)$.

Finally, $Q : q \to \mathbb{R}$, where $q = \bigcup_{e \in E} q_e$, is a function that maps quality levels q_e to a cost in order to express the monetary impact of the edge quality states. This cost is imposed by the asset manager on the service provider responsible for the edge to incentivise higher quality levels.

4.2 Maintenance Planning

In our problem we are dealing with infrastructural networks that are not serviced by the asset manager himself. Instead, maintenance is done by a set of N service providers (agents), each having a unique set of maintenance activities[6] A_i at their disposal, requiring unit time to perform. Each task $a \in A_i$ is defined by the tuple $\langle e, q', \alpha \rangle$, containing the edge e that is serviced, the effect q' on the quality and a success rate $\alpha \in [0, 1]$. In addition, agents have a private cost function $c_i : A_i \times T \to \mathbb{R}$ that for each of their tasks captures the cost of performing it at any time step.

Given their activities and cost function, the goal of the agents is to develop a maintenance plan P_i. This plan is a sequence of tasks $\{a_0, a_1, \ldots, A_{|T|-1}\}$ chosen from $A_i \cup \{\circ\}$, with $\circ$ being a no-operation task. The cost of such a plan for the agent is simply the sum of task costs, or $C_i(P_i) = \sum_{t \in T} c_i(P_i(t), t)$.

[5] The model does not depend on the actual choice for quality level representation, we could use for instance discrete levels, a normalised relative factor [0, 1], etc.

[6] We use the terms activities and tasks interchangeably.

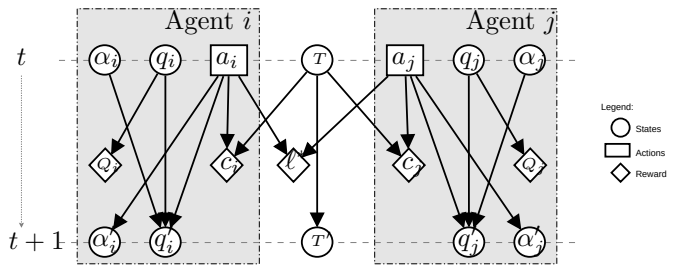

Figure 1 – A two-player example that represents two time steps of the MDP formulation for the maintenance planning problem.

We are interested in finding jointly *efficient*[7] plans that minimise the total cost of performing maintenance on the infrastructural network – i.e. task costs, quality costs and the (increase in) user costs – and not just the agent's individual cost. Simply solving these individual planning problems and combining them to form the joint plan $P = \bigcup_{i \in N} P_i$ neglects both the network quality as well as the impact the joint maintenance plan has on the network user. Denoting the task cost of the joint plan by $C(P) = \sum_{i \in N} C_i(P_i)$, then we are interested in the solutions: $P^* = \arg\min_{P \in \mathcal{P}} \; C(P) + \sum_{t \in T} \left(Q(q^t) + \ell^*(f^t(P)) \right)$ in which q^t is the vector of edge qualities updated according to the chosen activities (and their success), $f^t(P)$ the flow resulting from the planned tasks and $\mathcal{P}$ the set of all possible plans.

However, as discussed before, finding a single joint plan does not guarantee efficiency in contingent environments. Each task has a success rate $\alpha \in [0, 1]$ such that they can fail with some random probability $1 - \alpha$ and we need to react to such possible failures. Therefore we apply a dynamic mechanism, presented in the next sub section.

Note that other sources of uncertainty could be the task duration or the actual quality of an edge, but this simple modification allows us to study the complex dynamics of a uncertain world, while keeping the model conceptually easy .

4.3 Efficient Maintenance Planning Mechanism

Before we can apply a mechanism, we first need to formulate the planning problem as a game. The players (service providers) are modelled as rational, self-interested and autonomous agents that want to maximise their own utility. Each of the agents have a type space Θ_i that defines its local set of MDP states S_i. From this they choose for each time step t a type $\theta_i^t = \langle s_{\theta_i^t}, r_{\theta_i^t}, \tau_{\theta_i^t} \rangle$, with $s_{\theta_i^t}$ representing the current state, r the reward function and τ the transition function as before.

Instead of presenting the complex MDP formulation, we illustrate our model with a two-agent example in Figure 1.

In this figure we have depicted the state variables, the chosen actions and their dependencies for two time steps.[8] Also, the MDP rewards for the choices made in time t are illustrated by the diamonds in between the two states.

[7] The term efficient solution is commonly used in game theory for maximising overall utility, similar to our goal of minimising maintenance cost for all players.

[8] Note that the notation is not entirely correct, however to keep the figure readable we subscript the action, quality and success rate with the corresponding player index.

Agent utilities in dynamic mechanism design do not only depend on the valuation of the current outcome but also on the (expected) valuation of future outcomes. In order to compute its utility, agents also need information about possible future outcomes and strategies played by others. This functions is defined as $u_i(\theta^t, \pi, \sigma) = V_i(\theta_i^t, \pi, \sigma) - \mathcal{T}_i^*(\theta_i^t, \pi, \sigma)$ in which π is the policy decision function of the mechanism that maps types to outcomes, σ the set of strategies players use from the set of available strategies Σ, V_i the player valuation function and $\mathcal{T}_i^*$ the expected payment that this player has to pay to or receives from the center. Note that both the valuation and payment are not computed over a single type; instead, they are (possibly discounted) expectations over future outcomes given the policy and agent strategies. Agent valuations are defined as $V_i(\theta^t, \pi, \sigma) = \mathbb{E}\big[\sum_{k=t}^{T} \gamma^{k-t} r_i\big(\theta_i^k, \pi(\sigma(\theta^k))\big)\big]$ given the discount factor $\gamma \in [0, 1]$ and their MDP reward function r_i.

From Figure 1 we can see that each agent's reward function consists of the quality cost Q_i and its private cost c_i component. The revenue for doing their work is not included as part of the reward. This is because we assume w.o.l.g. that they will receive a fixed, contracted price and therefore only the cost components have to be optimised.

In addition to the service providers – the players we discussed so far – we also include the asset manager as a player, denoted by ϕ. The asset manager can be seen as the center of the mechanism, representing the network authority and acting on the behalf of the network users. The type space Θ_ϕ of the center contains the infrastructural network $\mathcal{N}$ and each type $\theta_\phi^t \in \Theta_\phi$ includes the network quality vector q^t and the current flow f^t. The asset manager cannot take actions, but its valuation function accounts for the network quality and user costs. Its value function V_ϕ is given by $V_\phi(\theta^t, \pi, \sigma) = \mathbb{E}\big[\sum_{k=t}^{T} \gamma^{k-t} \ell^*\big(\pi(\sigma(\theta^k))\big)\big]$, thus the expected quality and user cost given the joint type that results from policy π when players employ strategies σ.

If we include the center in the set of players, i.e. $N^* = N \cup \{\phi\}$, then we can define a dynamic-VCG mechanism for the maintenance problem. At each time t players report their private information θ_i^t to the center, who is able to develop efficient maintenance plans using the policy that can be computed using e.g. the centralised method presented in sub Section 2.1. I.e. the asset manager combines the local MDP's, finds the optimal policy and dictates the actions to be taken by the service providers.

5 Experiments

In order to compare and validate the dynamic mechanism we propose, we have implemented all of the algorithmic approaches presented in Section 2 and tested them on a set of 60 generated, two-player maintenance planning problem (Section 4) instances. These instances are generated using the following settings:

- Each player is responsible for one road only, with a discrete quality between 0 and 6. The quality cost function is linearly decreasing in the quality by some random factor from $[1, 3]$.

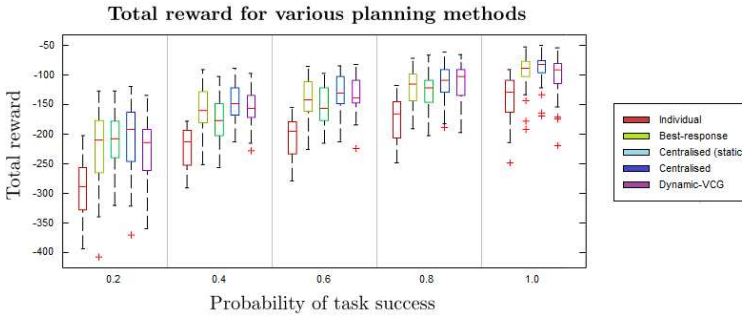

Figure 2 – Total plan rewards for various planning methods. With a 100% success rate, the static centralised and dynamic centralised algorithm are equivalent.

- Players have equi-sized task sets of size 1, 2 or 3 (excluding the no-op). Of each task set size we generated 20 instances. Tasks increase the quality level by 1, 2 or 3 units (corresponding to the task set sizes, i.e. a task set of size 2 contains a 1 and 2 unit effect task) and have a linear cost in the effect with a random factor from $[1, 3 * \text{effect}]$.
- The user cost is given by a penalty that is only awarded when both players perform an activity at the same time. The penalty value differs for the two players and is chosen randomly from an interval $[1, 10]$.

We ran all four algorithms on these 60 instances with different levels for the task success rates using a mixed-integer encoding. In addition, we also included the 'static' centralised algorithm that develops a plan in advance but does not respond to any task failures. Note that we used the same random seed for all methods so that the same contingencies occur.

5.1 Discussion

In the context of this paper we have chosen to include only the total plan reward results, depicted in Figure 2. From the efficiency perspective, the centralised algorithm can be considered the optimal method. The individual planner performs rather weak efficiency-wise as expected. The best-response method performs remarkably well (although taking a lot of time, already for the 3 iteration rounds we used) and is able to produce near-optimal plans.

Finally, we can see that the total rewards of our mechanism are close to optimal (i.e. compared to the centralised algorithm). Nonetheless, when no failures occur the mechanism produces rather expensive plans compared to the centralised method. With a low rate of success, the mechanism does not perform well. This is due to the fact that under high task failure rates, players have to make the same payments for the user cost they would cause to others again when rescheduling the task at a later time.

6 Conclusion

We presented several known algorithmic approaches and a novel dynamic mechanism for solving multi-agent planning under uncertainty. The main advantage of the mechanism is that under the mechanism players are compensated for their contribution towards efficient plans. Under the dynamic-VCG payments players have a higher utility for globally efficient plans than plans that minimise their own private cost. Our experiments support the application of such a mechanism to this domain.

Regretfully, the dynamic-VCG mechanism suffers a few weaknesses that may render it quite impractical for real-world application. Most notably the mechanism requires the participating agents (commercial service providers) to disclose all of their private information concerning maintenance activities and costs. In the competitive setting in which these agent operate, this might be unacceptable and therefore future research should be directed towards indirect dynamic mechanisms.

Acknowledgements

This research is part of the Dynamic Contracting in Infrastrures project and is supported by Next Generation Infrastructures and Almende BV. Matthijs Spaan is funded by the FP7 Marie Curie Actions Individual Fellowship #275217 (FP7-PEOPLE-2010-IEF).

References

Athey, S., I. Segal (2007) An efficient dynamic mechanism, Tech. rep., UCLA Department of Economics.

Bergemann, D., J. Valimaki (2006) Efficient dynamic auctions, *Cowles Foundation Discussion Papers*.

Cavallo, R. (2008) Efficiency and redistribution in dynamic mechanism design, in: *Proceedings of the 9th ACM conference on Electronic commerce*, ACM, pp. 220–229.

Dash, R., N. Jennings, D. Parkes (2003) Computational-mechanism design: A call to arms, *IEEE intelligent systems*, pp. 40–47.

Dean, T., L. Kaelbling, J. Kirman, A. Nicholson (1993) Planning with deadlines in stochastic domains, in: *Proceedings of the eleventh national conference on Artificial intelligence*, vol. 574, Citeseer, p. 579.

Gibbard, A. (1973) Manipulation of voting schemes: a general result, *Econometrica*, 41(4), pp. 587–601.

Hoey, J., R. St-Aubin, A. Hu, C. Boutilier (1999) Spudd: Stochastic planning using decision diagrams, in: *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann Publishers Inc., pp. 279–288.

Jonsson, A., M. Rovatsos (2011) Scaling up multiagent planning: A best-response approach, in: *Twenty-First International Conference on Automated Planning and Scheduling*.

Moonen, J. (2009) *Multi-agent systems for transportation planning and coordination*, Erasmus University Rotterdam.

Narahari, Y., D. Garg, R. Narayanam, H. Prakash (2009) *Game theoretic problems in network economics and mechanism design solutions*, Springer-Verlag New York Inc.

Nisan, N. (2007) *Algorithmic game theory*, Cambridge Univ Pr.

Pavan, A., I. Segal, J. Toikka (2008) Dynamic mechanism design: Revenue equivalence, profit maximization and information disclosure.

Seuken, S., R. Cavallo, D. C. Parkes (2008) Partially-synchronized dec-mdps in dynamic mechanism design, in: *Proceedings of the 23rd national conference on Artificial intelligence*, vol. 1, pp. 162–169.