Scaling Up Multiagent Planning: A Best-Response Approach

Anders Jonsson
Dept. Tecnologies de la Informació i les Comunicacions
Universitat Pompeu Fabra
08018 Barcelona, Spain

Michael Rovatsos
School of Informatics
The University of Edinburgh
Edinburgh EH8 9AB, United Kingdom

Abstract

Multiagent planning is computationally hard in the general case due to the exponential blowup in the action space induced by concurrent action of different agents. At the same time, many scenarios require the computation of plans that are strategically meaningful for self-interested agents, in order to ensure that there would be sufficient incentives for those agents to participate in a joint plan. In this paper, we present a multiagent planning and plan improvement method that is based on conducting iterative best-response planning using standard single-agent planning algorithms. In constrained types of planning scenarios that correspond to congestion games, this is guaranteed to converge to a plan that is a Nash equilibrium with regard to agents' preference profiles over the entire plan space. Our empirical evaluation beyond these restricted scenarios shows, however, that the algorithm has much broader applicability as a method for plan improvement in general multiagent planning problems. Extensive empirical experiments in various domains illustrate the scalability of our method in both cases.

Introduction

Multiagent planning, i.e. planning in the presence of multiple agents, has been a long-standing concern in the multiagent systems community (Durfee 1999), as the coordination of individual agents' planning processes is a hard problem in systems that allow concurrent action between autonomous, rational agents. While much of the research in the area has focused on managing ongoing collaborative activity at run-time (Durfee and Lesser 1991), defining communication mechanisms for collaborating planners (Durfee, Lesser, and Corkill 1985; Grosz and Kraus 1996), and dealing with the relationships between individual agents' local plans (Cox and Durfee 2005; Cox, Durfee, and Bartold 2005; Witteveen et al. 2005; Dimopoulos and Moraitis 2006), other work focuses more on the core algorithmic problems created by the presence of multiple agents during offline plan synthesis itself. In terms of dealing with distributed actions in purely *cooperative* domains, Boutilier and Brafman (2001) investigated the semantics of concurrent actions in partial-order planning, Brafman and Domshlak (2008) introduced a novel notion of loosely coupled agents that limits the combinatorial blow-up of planning complexity caused by multiple agents, and developed a more scalable centralised multiagent planner based on this concept (Nissim, Brafman, and Domshlak 2010). On the *strategic* side of multiagent planning, where self-interested agents want to achieve their local goals but depend on each others' actions, early attempts used social choice theory during the synthesis of multiagent plans (yet without using scalable planners) (Ephrati and Rosenschein 1994) tried to model notions of equilibria by analysing the relationships between different solutions in game-theoretic terms (Bowling and Veloso 2003; Larbi, Konieczny, and Marquis 2007). Others, more recently, focused on a more computational view of "planning games" (Brafman et al. 2009; 2010).

As has become clear from this body of research, two fundamental problems underlie the design of algorithms to construct multiagent plans: Firstly, the exponential blowup in the action space induced by allowing concurrent, independent action. And, secondly, in settings which involve higher levels of agent autonomy, additionally satisfying strategic side-conditions such as individual rationality to provide sufficient incentives for agents to participate in a joint plan. This makes it inherently hard to apply state-of-the-art single-agent algorithms, which have been shown to scale well in many complex domains in practice, directly to multiagent problems. Therefore, it is not surprising that none of the aforementioned approaches has attempted this.

In this paper, we present a multiagent planning and plan improvement method based on conducting iterative *best-response planning* steps, where each individual planning procedure only uses standard off-the-shelf (single-agent) planning technology. Our method is based on starting from an arbitrary initial joint plan, and then iterating over all agents repeatedly using cost-optimal planning in such a way that each agent optimises its "reaction" to the current set of other agents' local plans. We suggest a translation of the multiagent planning process to a transformed single-agent problem that enables use of standard planners, while ensuring that the resulting global plan is still valid (i.e. other agents' plans remain executable). At the same time, this calculation of local best responses avoids exponential blowup despite allowing us to define arbitrary interactions between individual agents' activities.

The contribution of our work is threefold: Firstly, we present a novel *best-response planning* method for general iterative plan improvement starting from arbitrary multiagent plans, and using standard planners. Secondly, we show how in a specific class of planning problems that correspond to congestion games, this method enables fast and scalable computation of equilibria for very large game instances. We are interested in such equilibria because of their compatibility with agents' self-interest, in the sense that no rational agent would have an incentive to deviate from such a joint plan unilaterally. In contrast to (Brafman et al. 2009; 2010), this follows the tradition of *non-cooperative* game theory, where it is not assumed that agreements among agents can be enforced, and every agent can only adapt locally to what other agents are doing.

Thirdly, we show that despite the absence of convergence or optimality guarantees, our method proves to be useful for improving general multiagent plans in scenarios in which an initial plan is available but could be further optimised.

We proceed as follows: We first introduce the overall problem and the suggested solution method, illustrating these with a simple example. Then, we apply our model to congestion games (Rosenthal 1973; Monderer and Shapley 1996), and show how a slightly relaxed form of cost function satisfies the necessary properties and gives us more flexibility in the kinds of planning problems for which we can derive Nash equilibria. An extensive empirical evaluation in the subsequent section illustrates the benefits of our method in congestion planning and other multiagent planning problems. We close with some conclusions and an outlook on future avenues of research on the topic.

Multiagent Planning Problems

Our planning formalism is based on a STRIPS-style model of classical planning under full observability. States are represented by sets of ground fluents taken from a set F, and actions are given as tuples $a = \langle \mathrm{pre}(a), \mathrm{eff}(a) \rangle$, where $\mathrm{eff}(a)$ contains positive fluents p to be added to the state and negative fluents $\neg p$ to be deleted from the state after execution of a. For a state $s \in 2^F$, action a can only be executed if $\mathrm{pre}(a) \subseteq s$, and will result in state

$$\theta(s,a) = (s \cup \{p | p \in \mathrm{eff}(a)\}) \setminus \{p | \neg p \in \mathrm{eff}(a)\}.$$

While this definition does not include negative preconditions, it is easy to accommodate them if needed. Our formulation of a *multiagent planning problem* (MAP) borrows heavily from MA-STRIPS (Brafman and Domshlak 2008) and coalition-planning games (Brafman et al. 2009). It considers the agents' individual goals and cost functions (defined over the *joint* action space) as independent, but assumes that the individual reward for an agent is entirely captured by the cost function. That is, it does not introduce any extra reward associated with goal achievement; if this reward is identical for all goal-achieving states and zero else, rewards for goal achievement do not add to the overall expressiveness of the problem. We define a MAP as a tuple $\Pi = \langle N, F, I, \{G_i\}_{i=1}^n, \{A_i\}_{i=1}^n, \Psi, \{c_i\}_{i=1}^n \rangle$, where

- $N = \{1, \ldots, n\}$ is the set of agents,
- F is the set of fluents,
- $I \subseteq F$ is the initial state,
- $G_i \subseteq F$ is agent i's goal,
- A_i is agent i's action set,
- $\Psi : A \rightarrow \{0,1\}$ is an admissibility function,
- $c_i : \times_{i=1}^n A_i \rightarrow \mathbb{R}$ is the cost function of agent i.

We write $A = A_1 \times \ldots \times A_n$ for the joint action set assuming a concurrent, synchronous execution model, and $G = \wedge_i G_i$ for the conjunction of all agents' individual goals.

A MAP typically imposes concurrency constraints regarding actions that cannot or have to be performed concurrently by different agents to succeed (Boutilier and Brafman 2001). Instead of representing the set of joint actions explicitly, the admissibility function Ψ indicates whether a joint action $a \in A$ is part of the MAP ($\Psi(a) = 1$) or not ($\Psi(a) = 0$). Although the number of joint actions is generally exponential in the number of agents, Ψ can usually be represented much more compactly. Crucially, our approach requires being able to check whether a joint action is part of the MAP in a fast way, which is another motivation for including Ψ.

A *plan* $\pi = \langle a^1, \ldots, a^k \rangle$ is a sequence of joint actions $a^j \in A$ such that a^1 is applicable in the initial state I (i.e. $\mathrm{pre}(a^1) \subseteq I$), and a^j is applicable following the application of $a^1, \ldots, a^{j-1}$ (i.e. $\mathrm{pre}(a^j) \subseteq \theta(I, \langle a^1, a^2, \ldots, a^{j-1} \rangle)$, where θ is canonically extended to sequences of actions), for all $2 \leq j \leq k$. We say that π *solves* the MAP Π if the goal state G is satisfied following the application of all actions in π, i.e. $G \subseteq \theta(\pi)$. The cost of a plan π to agent i is given by $C_i(\pi) = \sum_{j=1}^k c_i(a^j)$.

Before describing our method for best-response planning in MAPs, we need to introduce a few more auxiliary definitions: To start off with, following Brafman and Domshlak (2008), we can partition the set of fluents into

$$F = F_1 \uplus \ldots \uplus F_n \uplus F_{pub}$$

using the usual distinction between public and private fluents. For this, let $F(A_i) = \cup_{a \in A_i} (\mathrm{pre}(a) \cup \mathrm{eff}(a))$ be the set of all fluents appearing in A_i. The *private fluents* of agent i are defined as

$$F_i = F(A_i) \setminus (\cup_{j \neq i} F(A_j))$$

and capture the fluents entirely under i's control. This leaves

$$F_{pub} = F \setminus (\cup_i F_i)$$

as the set of remaining *public* fluents, over which at least two agents interact. It follows directly from this distinction that each action $a_i \in A_i$ is restricted to have preconditions and effects on the fluents in $F_i \cup F_{pub}$ only. Note also that the agents' individual goals G_i are not independent, as they may interact over public fluents.

Furthermore, we define the preconditions and effects of a joint action $(a_1, \ldots, a_n) \in A$ as the union of the preconditions and effects of its constituent single-agent actions a_i. We use the notation $a = (a_i, a_{-i})$ which involves the shorthand a_{-i} to denote the joint action of all agents but i.

Finally, for reasons which will become obvious below, we add a no-op action $noop_i$ with no effects to the action set A_i of each agent i. Our intention is that an agent should always be able to select the no-op action once it is done with its other actions, but depending on the problem an agent may also be allowed to choose it in the middle of a plan. We define the cost to agent i of the no-op action as $c_i(noop_i, a_{-i}) = 0$ for any a_{-i}.

Best-Response Planning

In principle, solving a MAP involves an action space that grows exponentially in the number of agents involved. Due to the limited scalability of cost-optimal planning, this issue is particularly problematic when attempting to do centralised planning with full concurrency and to minimise an overall cost measure (e.g. social welfare $\sum_i C_i(\pi)$). In more strategic settings, a MAP algorithm has to additionally respect criteria such as individual rationality to provide an incentive for self-interested agents to participate (Brafman et al. 2009; 2010). This makes the plan search problem harder still, as a "solution" in this setting can only be a plan that is compatible with all agents' preferences and robust against individual deviation, e.g. a plan that is a Nash equilibrium.

Our method aims at tackling both these issues, by iteratively solving simpler *best-response planning* (BRP) problems from the point of view of an individual agent i. The basic idea is that given some current plan π^k which solves a MAP Π, agent i calculates a plan π^{k+1} which minimises i's cost while only modifying i's actions in π^k:

$$\pi^{k+1} = \arg\min\{C_i(\pi) | \pi \text{ identical to } \pi^k \text{ for all } j \neq i\}$$

For this purpose, given a joint plan π of length k that solves Π, we define a BRP problem with costs as $\Pi_i = \langle F', I', G', A'_i, c' \rangle$, where

- $F' = F_i \cup F_{pub} \cup \{\mathtt{time}(0), \ldots, \mathtt{time}(k)\}$,
- $I' = (I \cap F') \cup \{\mathtt{time}(0)\}$,
- $G' = (G \cap F') \cup \{\mathtt{time}(k)\}$,
- For each a^j_{-i} and $a_i \in A_i$, construct the joint action $a = (a_i, a^j_{-i})$. If $\Psi(a) = 1$, add an action a' to A'_i with
  - $\mathrm{pre}(a') = (\mathrm{pre}(a) \cap F') \cup \{\mathtt{time}(j-1)\}$
  - $\mathrm{eff}(a') = (\mathrm{eff}(a) \cap F') \cup \{\mathtt{time}(j), \neg\mathtt{time}(j-1))\}$
  - $c'(a') = c_i(a)$.
- For each action $a_i \in A_i$, add an action a'' to A'_i with
  - $\mathrm{pre}(a'') = \mathrm{pre}(a_i) \cup \{\mathtt{time}(k)\}$
  - $\mathrm{eff}(a'') = \mathrm{eff}(a_i)$
  - $c'(a'') = c_i(a_i, noop_{-i})$.

This construction is based on three ideas: Firstly, agent i only has to worry about satisfying the goal state on fluents in $F_i \cup F_{pub}$, since it has no influence over the remaining fluents. This allows us to ignore all fluents F_j for $j \neq i$ in F', I' and G'.

Secondly, agent i is only provided with actions in step j that include the preconditions and effects of the actions taken by other agents in π, i.e. it is forced to pick only actions that comply with other's "fixed" actions in terms of $F_i \cup F_{pub}$. Thereby, additional fluents $\mathtt{time}(0), \ldots, \mathtt{time}(k)$ are used to ensure that all joint actions of other agents (a^1_{-i} through a^k_{-i}) are executed at the right time.

Thirdly, the individual agents' plans may have different lengths, which is accounted for by no-op actions. Recalling that $noop_j$ is always applicable whenever agent j is done with its other actions, we allow agent i to extend the plan π with additional joint actions a'' that correspond to all other agents performing $noop_j$ actions. Since such actions a'' are only admissible after the first k joint actions (due to the precondition $\mathtt{time}(k)$), the no-op action $noop_j$ is always applicable for each agent $j \neq i$ and does not alter its other action choices. The cost of these actions a'' is 0 to each agent $j \neq i$ by definition, while the cost to agent i reflects the cost of its action a_i when all other agents are inactive.

Theorem 1 *Given a joint plan π that solves Π and a solution π_i to the BRP problem Π_i, the joint plan π' that results from replacing the actions of agent i in π with π_i solves Π.*

Proof Replacing the actions of agent i does not affect fluents in F_j, $j \neq i$. Since the goal on those fluents is satisfied by π, they are satisfied by π' also. The goal on the remaining fluents $F_i \cup F_{pub}$ is satisfied by π_i. Moreover, π_i ensures that the pre-conditions on public fluents of other agents' actions are satisfied. Thus π' solves Π.

With these provisions made, we can define a *best-response planner* for agent i as an algorithm which, given a solution π^k to a MAP Π finds a solution π^{k+1} to the transformed planning problem Π_i with minimum cost $C_i(\pi^{k+1})$ among all possible solutions. Since Π_i is a single-agent planning problem, any cost-optimal planner can be used as a best-response planner. We borrow the term "best response" from game theory here in the following sense: Assume each agent's contribution to a plan π is denoted by π_i (a sequence of $a_i \in A_i$), with π_{-i} the joint plan of all remaining agents. If we define the utility of agent i as

$$u_i(\pi_i, \pi_{-i}) = \begin{cases} -C_i(\pi_i, \pi_{-i}) & \text{if } (\pi_i, \pi_{-i}) \text{ is feasible} \\ -\infty & \text{else} \end{cases}$$

and π_i is the solution to Π_i based on π as above, then π_i is a best-response strategy for i in the game-theoretic sense.

Example

To illustrate the suggested transformation, consider the network routing example shown in Figure 1. Three packets (agents) *1*, *2*, and *3* are using links between nodes in a grid-like network to move from origin to destination. We assume that the cost for each agent to move across a link equals the number of agents using it concurrently in the same time step.

First, consider a situation with no public fluents F_{pub}, i.e. every agent can use any local action independently without any side-effects on others except a potential effect on cost. We have fluents $\mathtt{at}(i, n)$ for agent i being located in node n, and $\mathtt{link}(l, n_1, n_2)$ to represent links (two for each link to capture both directions). The only (single-agent) action for agent i is $move(i, l, n_1, n_2) = \langle \{\mathtt{at}(i, n_1), \mathtt{link}(l, n_1, n_2)\}, \{\neg\mathtt{at}(i, n_1), \mathtt{at}(i, n_2)\} \rangle$, and $G_i = \{\mathtt{at}(i, n_i)\}$ for some target node n_i.

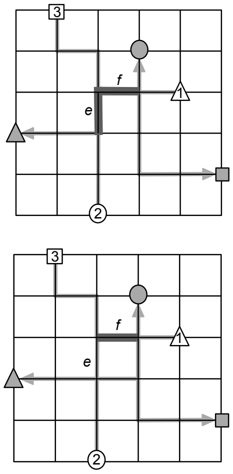

Figure 1: A routing example. Targets of packets are shown as shaded shapes, paths as shaded arrows. Edges concurrently used by more than one agent (e, f) are shown with a thicker shading.

Starting from an initial plan as shown in the top figure, where agents *1* and *2* share link e (at time step 3) and agents *2* and *3* share link f (at time step 4), we sketch the BRP step of agent *1*. As actions a' in our definition above, we add new $move$-actions of the form

$$\langle\{\texttt{at}(\mathit{1},n_1),\texttt{link}(l,n_1,n_2),\texttt{time}(j-1)\},$$
$$\{\neg\texttt{at}(\mathit{1},n_1),\texttt{at}(\mathit{1},n_2),\neg\texttt{time}(j-1),\texttt{time}(j)\}\rangle$$

for each pair of nodes n_1, n_2 connected by a link l and each $j \in \{1,\dots,k\}$, where $k = 8$ is the number of actions in the longest (*3*'s) plan. Here, we need not consider any new preconditions or effects other than `time`, as there are no (other) public fluents, but when $j = 3$ holds we would have to set $c'(a') = 2$ for the new $move$-action along link e, since this link is used by agent *2* at that time step, and this would be also the case for any other joint actions in which other agents were already using links. In the process of cost-optimal planning, agent *1* will avoid e as alternative links are available that have cost of 1 rather than 2, resulting in the situation depicted in the bottom figure.

Now, to illustrate the use of public fluents, consider a slightly different situation, where agents can switch links on or off. We can model this using public fluents $\texttt{on}(l)$ such that each move action along a link l has an additional precondition $\texttt{on}(l)$. Each agent has two additional actions to switch links on and off:

$$\langle\{\texttt{at}(i,n_1),\texttt{link}(l,n_1,n_2),\neg\texttt{on}(l)\},\{\texttt{on}(l)\}\rangle,$$
$$\langle\{\texttt{at}(i,n_1),\texttt{link}(l,n_1,n_2),\texttt{on}(l)\},\{\neg\texttt{on}(l)\}\rangle.$$

If links are initially turned off, agents have to first turn them on before sending packets. Given an initial joint plan such that agent *1* switches on a link l' later used by agent *2*, each action a' of the BRP problem of agent *1* now includes preconditions on the public fluents $\texttt{on}(l)$ of other agents' actions. A plan solving the BRP problem cannot deviate by not switching on l', since that would fail to satisfy the precondition $\texttt{on}(l')$ of each action a' corresponding to the time step j for which agent *2* sends a packet across link l'.

Regarding the "postfix" actions a'' added in our transformation, consider how agent *3* computed its plan before the situation shown in the top figure: Since agents *1* and *2* had plans of length 5, we add another $move$-action with an additional precondition $\texttt{time}(5)$. Beyond this time step, time does not progress in terms of fluents, and the actions have cost independent of others' actions as they are now inactive.

Congestion Planning

In the general case, iterative BRP with all agents taking turns starting from an initial solution π^0 to the MAP Π is not guaranteed to converge. However, it is known from game theory that iterative BRP will converge to a pure-strategy (i.e. deterministic) Nash equilibrium for a class of games known as *congestion games* (Rosenthal 1973; Monderer and Shapley 1996). In this section, we will define a class of *congestion planning problems*, which we show to satisfy the definition of congestion games, such that the procedure defined above is guaranteed to converge.

We start by reviewing the game-theoretic definition and then introduce planning problems that are closely related to congestion games. A *congestion game* is a tuple $\langle N, R, A, c\rangle$, where

- $N = \{1,\dots,n\}$ is a set of agents,
- $R = \{r_1,\dots,r_m\}$ is a set of resources,
- $A = A_1 \times \ldots \times A_n$, where $A_i \subseteq 2^R - \emptyset$ is the action set of agent i,
- $c = (c_{r_1},\dots,c_{r_m})$, where $c_r : \mathbb{N} \to \mathbb{R}$ is the cost function of resource r.

In such games, each agent acts by selecting a non-empty subset of resources. The utility function $u_i : A \to \mathbb{R}$ of agent i is defined as

$$u_i(a) = -\sum_{r\in a_i} c_r(\#(r,a)), \tag{1}$$

where a_i is the action choice of i in the joint action a and $\# : R \times A \to \mathbb{N}$ is a counting function that returns the number of agents in N that select resource $r \in R$ in $a \in A$.

Given a congestion game, we can define a *potential function* $Q : A \to \mathbb{R}$ as

$$Q(a) = \sum_{r\in R}\sum_{j=1}^{\#(r,a)} c_r(j). \tag{2}$$

For two joint actions (a_i, a_{-i}) and (a'_i, a_{-i}) that only differ in the action choice of agent i, it is possible to show that $Q(a_i, a_{-i}) - Q(a'_i, a_{-i}) = u_i(a_i, a_{-i}) - u_i(a'_i, a_{-i})$. A game with this property is known as a *potential game* (Monderer and Shapley 1996). If agent i acts greedily to increase its utility given a fixed action choice of the other agents, the potential increases by an equivalent amount. Thus best-response planning will always have the effect of increasing the overall potential and, if repeated, this process will eventually converge to a pure-strategy Nash equilibrium.

To exploit this property in a planning setting, we first make the utility function more interesting from a planning perspective. For each $i \in N$, we add a term to the utility function u_i, obtaining a new utility function

$$u'_i(a) = u_i(a) - d_i(a_i), \qquad (3)$$

where $d_i : A_i \to \mathbb{R}$ depends on the action choice $a_i \in A_i$ of agent i only. We also define an associated potential function

$$Q'(a) = Q(a) - \sum_{j \in N} d_j(a_j). \qquad (4)$$

It is easy to prove that this is still a potential game:

$$\begin{aligned} Q'(a_i, a_{-i}) \; - & Q'(a'_i, a_{-i}) \\ = \quad & Q(a_i, a_{-i}) - d_i(a_i) - \sum_{j \in N - \{i\}} d_j(a_j) \\ & -Q(a'_i, a_{-i}) + d_i(a'_i) + \sum_{j \in N - \{i\}} d_j(a_j) \\ = \quad & Q(a_i, a_{-i}) - Q(a'_i, a_{-i}) - d_i(a_i) + d_i(a'_i) \\ = \quad & u_i(a_i, a_{-i}) - u_i(a'_i, a_{-i}) - d_i(a_i) + d_i(a'_i) \\ = \quad & u'_i(a_i, a_{-i}) - u'_i(a'_i, a_{-i}). \end{aligned}$$

With this, we can define a *congestion planning problem* as a MAP Π by introducting a set of resources $R = \{r_1, \ldots, r_m\}$ and, for each resource $r \in R$, a cost function $c'_r : \mathbb{N} \to \mathbb{R}$. Each action $a_i \in A_i$ of agent i uses a (possibly empty) set of resources $R(a_i) \subseteq R$. The following additional restrictions are imposed on Π:

- $F_{pub} = \emptyset$.
- $\Psi(a) = 1$ for any $a \in A$.
- The cost function of agent i is of the form $c_i(a) = \sum_{r \in R(a_i)} c'_r(\#(r, a)) + d_i(a_i)$, where $d_i : A_i \to \mathbb{R}$ depends on the action choice a_i of agent i only.
- A no-op action uses no resources and incurs no cost, i.e., $R(noop_i) = \emptyset$ and $d_i(noop_i) = 0$.

Note that for each joint action $a \in A$ and each agent i, it holds that $c_i(a) = -u'_i(a)$, where u'_i is the modified utility function for congestion games from Equation (3). Thus, the modified potential function Q' in Equation (4) satisfies $Q'(a_i, a_{-i}) - Q'(a'_i, a_{-i}) = c_i(a'_i, a_{-i}) - c_i(a_i, a_{-i})$.

In a congestion planning problem, each agent i can plan individually to reach its goal $G_i = G \cap F_i$ from its initial state $I_i = I \cap F_i$. No other agent can contribute to achieving G_i since the fluents in G_i are private to agent i. However, there is still a loose interaction between agents since the cost of a plan to i depends on how many other agents use the same resources simultaneously.

Theorem 2 *For any congestion planning problem, best-response planning converges to a pure-strategy Nash equilibrium.*

Proof For each plan $\pi = \langle a^1, \ldots, a^k \rangle$, define a potential function $Q(\pi) = \sum_{j=1}^k Q'(a^j)$. Consider two plans $\pi = \langle a^1, \ldots, a^k \rangle$ and $\pi' = \langle a^{1'}, \ldots, a^{k'} \rangle$ that only differ in the action choices of agent i. We have

$$\begin{aligned} Q(\pi) - Q(\pi') &= \sum_{j=1}^k \left[Q'(a^j) - Q'(a^{j'}) \right] \\ &= \sum_{j=1}^k \left[c_i(a^{j'}) - c_i(a^j) \right] = C_i(\pi') - C_i(\pi). \end{aligned}$$

In other words, a congestion planning problem is a potential game such that if one agent greedily decreases its cost given a fixed selection of actions of the other agents, the potential increases by an equivalent amount.

Recall that our definition of best-response planning allows an agent i to extend a plan π with additional joint actions. We show that the above analysis holds in this case also. Let π' be the extended plan, i.e., π' includes m joint actions in addition to the k actions of π. Then we can extend π with m joint actions consisting of the no-op action for each agent, obtaining a new plan π''. Since no-op actions use no resources and incur no cost, it holds that $Q(\pi'') = Q(\pi)$ and $C_i(\pi'') = C_i(\pi)$. We can now use the same reasoning as above for π'' and π' since they have the same length and contain the same actions for agents different from i.

For a congestion planning problem Π, it is easy to come up with an initial plan π that solves Π: simply let each agent solve its part of the planning problem on its own, assuming that no other agents use the same resources simultaneously. Extend the plan of each agent with the no-op action until each individual plan has the same length. Finally construct π by joining the actions of each agent. By Theorem 2, starting with π and repeatedly performing best-response planning for each agent in turn is guaranteed to converge to a pure-strategy Nash equilibrium. The simple version of our network routing example in Figure 1 is actually a congestion planning problem, and illustrates the overall process.

Evaluation

We perform two sets of experiments to evaluate best-response planning empirically. In the first set of experiments, we test BRP in the network routing domain from our example, varying the number of nodes of the network as well as the number of agents. In the second set of experiments, we test BRP on the three multi-agent domains of Nissim, Brafman, and Domshlak (2010).

Network Routing Domain In the network routing domain, we randomly generate networks of given sizes, adding nodes to the network one at a time, connecting each node to k nodes in the existing network, with k a random number in the range $[1..M]$. This process guarantees that the resulting network is strongly connected. In the experiments we use $M = 3$, which ensures that packets have to travel along multiple links to reach their destination, while still allowing packets to choose between multiple possible routes.

For each network, we define a set of resources $R = \{l_1, \ldots, l_m\}$, where $l_1, \ldots, l_m$ are the links of the network. We associate a random capacity $C(l)$ in the range $[1..10]$ with each link $l \in R$. The cost of n agents simultaneously sending a packet across link l is defined as

$$c'_l(n) = L \cdot n + E^{\max\{0, n - C(l)\}}.$$

Agents	T	IC	FC	IT
10	33±17	35	33	2.4
20	87±50	75	71	3.2
30	226±126	128	112	3.4
40	293±322	177	154	3.6
50	334±180	224	192	3.6
60	335±53	280	229	4
70	502±236	353	270	4.2
80	721±263	416	315	4.6
90	836±442	484	368	4.6
100	669±156	595	435	4.4
110	1031±228	686	497	5
120	1041±244	783	540	5
130	1109±133	794	541	5
140	1696±389	944	648	5.4
150	2018±194	1006	699	5.2
160	3225±1592	1253	796	6
170	3874±1571	1366	862	7.2
180	2643±962	1427	879	6
190	3399±1235	1512	984	6.4
200	3484±801	1997	1124	6.4

Table 1: BRP in networks with 100 nodes and a varying number of agents, with regard to running time (**T**), initial cost (**IC**), final cost (**FC**), and number of iterations (**IT**).

The cost is linear as long as the number of packets does not exceed the link capacity, then becomes exponential – this roughly emulates models of congestion in computer networking. In the experiments we assume $L = 1$ and $E = 2$.

For each network we define a congestion planning problem as a MAP augmented with the set of resources R and the cost functions $c' = (c'_{l_1}, \ldots, c'_{l_m})$. Each agent has to send its packet between two distinct nodes of the network, drawn at random. We allow multiple agents to send packets between the same pair of nodes.

We use the HSP$^*_F$ planner (Haslum 2008) to solve each BRP problem optimally. Table 1 shows the results of running BRP in networks of 100 nodes, varying the number of agents in increments of 10. The results are averaged across 5 networks generated at random for a given number of agents. The table shows the running time in seconds (**T**), the total cost of the initial plan (**IC**), the total cost of the final plan (**FC**), as well as the number of BRP iterations (**IT**). The first thing to observe here is that in congestion planning problems, BRP scales to a number of agents well beyond the state-of-the-art in general multiagent planning.

We also tested how the number of network nodes influences running time. Table 2 shows the results of running BRP for 100 agents, varying the number of nodes in increments of 10. Here, the results are averaged across 10 networks generated at random. For a fixed number of agents, the smaller the network, the higher the cost of the initial plan, since there is more congestion in the network. We conclude that the more constrained the problem environment, the higher the relative benefit obtained from using BRP.

Nodes	T	IC	FC	IT
10	64±34	2828941	9344	7.4
20	101±38	16633	908	6.7
30	142±24	1320	575	5.7
40	319±161	911	528	5.9
50	306±73	795	472	5.4
60	477±195	692	439	5.8
70	548±102	605	420	5.1
80	853±415	615	425	4.9
90	1002±530	603	420	4.7
100	713±184	575	423	4.4

Table 2: BRP in networks with 100 agents and a varying number of network nodes.

IPC Domains We also evaluate BRP in more general MAPs, with two general objectives in mind: (1) to find out whether or not BRP converges in such problems, despite the lack of convergence guarantees, and (2) to assess the efficiency and effectiveness of BRP in MAPs with public fluents, i.e. when the interaction between agents is more complex. Unfortunately, there are very few general-purpose solvers for multiagent planning, making it difficult to come up with initial plans for such MAPs, which are a prerequisite for the BRP procedure to be applied.

One exception is the DisCSP solver of Nissim, Brafman, and Domshlak (2010), which uses distributed constraint satisfaction to coordinate the action selection of agents. To test the solver the authors reformulated three domains from the International Planning Competition (Logistics, Rovers, and Satellite) as MAPs. We use the DisCSP solver to come up with initial plans for various problems in these domains.

The IPC domains impose concurrency constraints on the joint actions (e.g. in Logistics, two trucks should not be able to simultaneously load the same package). We compactly represent the admissibility function Ψ in the following way: We first compute invariants of the problem, i.e. sets of fluents such that no more than one can be true at any moment. For each action of an agent with a public effect, we store the time step and the corresponding invariant. When generating the BRP problem associated with an agent, we disallow actions with public effects on an invariant already occupied by another agent at the same time step.

Our basic BRP problem formulation prevents joint plans from becoming shorter, since each agent has to compute a plan of length at least k, the current length of the joint plan. However, the above approach makes it possible to define k as the last time step for which the action of another agent has a pre-condition or effect on a public fluent. In practice, this makes it possible to decrease the length of the joint plan.

In each of the three domains we allow agents to choose no-op actions in the middle of a plan. For each action $a_i \in A_i$ different from the no-op action and each a_{-i}, we define $c_i(a_i, a_{-i}) = 1$. In other words, the cost of a plan to an agent equals the number of times the agent chooses an action different from the no-op action.

In addition to HSP$^*_F$, we implement BRP using LAMA (Richter and Westphal 2010), allowing us to test whether the convergence and quality of BRP suffer when using a satisficing planner. We allow LAMA to search for increasingly cheaper solutions until no more solutions are found. Since the cost 0 associated with no-op actions makes LAMA slow (likely due to heuristic plateaus) we associate a base cost with no-op actions. A base cost of 1 makes LAMA run much faster, but the quality of the solution suffers since LAMA does not distinguish between no-op and other actions. In the experiments we use a base cost of 0.5, which provides a decent tradeoff between speed and solution quality.

Table 3 shows the results of running BRP in the three IPC domains. The name of the problem indicates the number of agents and, for Logistics, the number of packages. The column DisCSP shows the running time of the DisCSP solver on each problem, as well as the total cost and makespan of the resulting solution. In each problem the DisCSP solver computes solutions with optimal cost, but since the solver applies actions with public effects in sequence, the makespan (i.e. length of the joint plan) is suboptimal in Logistics and Satellite.

The column BRP-Optimal shows the running time of iteratively applying BRP using HSP$^*_F$. The table also shows the number of iterations over all agents required to converge, as well as the cost and makespan of the resulting solution. The column BRP-Satisficing shows the same information when using LAMA to solve the BRP problems. In the largest Logistics problem (marked with an asterisk) the DisCSP solver is unable to come up with a solution, so we instead use LAMA on the original IPC problem to generate a sequential solution (which involves no concurrent actions at all).

In Logistics, BRP using HSP$^*_F$ quickly converges to an optimal solution with shorter makespan than the initial plan. This holds true even when the initial plan is purely sequential, suggesting that combining BRP with sequential planning may be a viable alternative in large domains. In contrast, LAMA reports that the corresponding BRP problems are unsolvable, even though HSP$^*_F$ solves them. The time predicate appears problematic to LAMA, which typically prefers to apply no-op actions until all other agents are done with their actions. Only in later iterations does LAMA find shorter concurrent plans, but in Logistics any plan has to be concurrent, which might be what causes LAMA to fail.

In Rovers, the DisCSP solver outputs plans with optimal cost and makespan, depriving BRP of the opportunity to improve on the initial plan. Still, BRP using LAMA converges and does not increase the overall cost even though optimality is no longer guaranteed. Interestingly, in Rovers the situation is reversed: HSP$^*_F$ reports that the BRP problems are unsolvable, even though LAMA solves them. We have no good explanation for this but are curious to find out why.

In Satellite, BRP using HSP$^*_F$ again converges to an optimal solution with shorter makespan than the initial plan. BRP using LAMA also converges and does not increase the overall cost. The difference in makespan between the two approaches is largely incidental: LAMA prefers the no-op actions at the beginning of the plan, while HSP$^*_F$ prefers them at the end. We could force planners to put no-op actions at the end by introducing two different no-op actions, one of which has lower cost but can only be applied once the agent is done with other actions. However, for ease of presentation we decided against such an approach in this paper.

The running time of BRP is not really comparable to that of the DisCSP solver since they solve two fundamentally different problems. However, the running time at least provides some indication of the relative complexity. In particular, the complexity of the DisCSP solver highly depends on the number of public fluents of the problem. As a result, it is quite efficient in Rovers and Satellite, which are domains with few public fluents. However, in Logistics, where there is more interaction between agents in the form of packages, the solver does not scale to problems with a number of agents and packages larger than a handful.

On the other hand, the complexity of BRP is not directly dependent on the number of public fluents. Instead, the complexity of optimal planning largely depends on the branching factor and the plan horizon. In Logistics, which has a fairly small branching factor, BRP is relatively fast. In Rovers and Satellite, the branching factor is larger, making it slower to solve the associated BRP problems optimally.

Conclusions

We have introduced best-response planning as a method for planning and plan improvement in multiagent planning problems with full concurrency. We suggested a transformation of the original MAP that allows individual agents to improve their local "response" to the behaviour of other agents in a current plan using off-the-shelf (single-agent) planners.

For planning problems that correspond to (a relaxed version of) congestion games, and for which local best-response search is known to converge to a Nash equilibrium, we were able to show that best-response planning allows the computation of equilibrium plans in multiagent domains that are clearly too large to be tackled with other existing MAP methods in a centralised and fully cooperative – let alone strategic i.e. incentive-compatible – way.

Since interaction between agents in these congestion planning problems is limited to effects on the cost incurred to each participating agent, congestion planning problems do not model domains in which an agent either needs other agents' actions to be able to achieve her goal, or domains in which certain individual actions can make the future execution of others' actions impossible. To assess the usefulness of the best-response planning in these cases, we tested it using optimal and satisficing planners in multiagent planning problems derived from IPC domains. Here, we saw that a few best-response iterations usually suffice to improve the initial plan computed by existing distributed planning methods when there is room for such improvement.

An advantage of our algorithm is that we need not make any assumptions as to how to the initial plan is computed in a multiagent system in terms of computational distribution. The agents could perform their own planning steps asynchronously and exchange information periodically using some mutual notification protocol, or they could rely on a central planning agent who executes the algorithm for all agents iteratively until convergence is achieved, or for a fixed number of iterations. This gives us a lot of flexibility regarding potential uses of our algorithm in multiagent systems. Further, as we only use standard planning technology, the performance of our algorithm will directly benefit from any future advances in the area of cost-optimal planning.

	DisCSP			BRP-Optimal				BRP-Satisficing			
Problem	**Time**	**Cost**	**MS**	**Time**	**Iterations**	**Cost**	**MS**	**Time**	**Iterations**	**Cost**	**MS**
Logistics_3_1	1.3	10	9	0.2	1	10	9	-	-	-	-
Logistics_4_2	307.0	14	12	0.6	3	14	6	-	-	-	-
Logistics_6_2*	126.7	20	20	5.3	3	20	9	-	-	-	-
Rovers_3	53.0	33	13	-	-	-	-	560.2	2	33	13
Rovers_4	408.4	44	14	-	-	-	-	936.6	2	44	14
Rovers_5	784.2	55	15	-	-	-	-	2170.7	3	55	15
Rovers_6	3958.7	66	16	-	-	-	-	2235.2	2	66	16
Satellite_2	0.5	7	4	0.2	2	7	4	0.8	2	7	4
Satellite_4	1.2	14	6	1.5	2	14	4	5.7	3	14	6
Satellite_6	3.4	21	8	19.4	2	21	4	13.5	2	21	8
Satellite_8	25.5	28	10	178.0	2	28	4	37.6	2	28	10

Table 3: Results of running BRP in the three IPC domains.

In the future, we would like to develop criteria for determining when a planning problem represents a congestion game, to investigate practical convergence properties of best-response planning across many more typical planning domains, and to assess the impact of using satisficing planners instead of optimal planners in this setting. Another question that requires further investigation is the quality of the equilibria computed, how it depends on the initial plan and on the order in which the BRP steps are performed when iterating over agents, and how overall system performance evolves across BRP steps in which agents effectively only greedily decrease their own cost. Finally, we would like to investigate the viability of using BRP or some variant thereof to come up with initial plans. This is challenging both in terms of individual agents solving part of the problem on their own, and synchronizing the plans of individual agents to make sure the resulting joint plan is coherent.

Acknowledgments

This work was partially supported by APIDIS and TIN2009-10232, MICINN, Spain. The authors also thank Raz Nissim for providing the source code of the DisCSP planner.

References

Boutilier, C., and Brafman, R. I. 2001. Partial-order planning with concurrent interacting actions. *Journal of Artificial Intelligence Research* 14: 105–136.

Bowling, M., and Veloso, M. 2003. Simultaneous adversarial multi-robot learning. In *Procs IJCAI 2003*, 699–704.

Brafman, R. I., and Domshlak, C. 2008. From One to Many: Planning for Loosely Coupled Multi-Agent Systems. In *Procs ICAPS 2008*, 28–35. AAAI Press.

Brafman, R. I.; Domshlak, C.; Engel, Y.; and Tennenholtz, M. 2009. Planning Games. In *Procs IJCAI 2009*, 73–78.

Brafman, R. I.; Domshlak, C.; Engel, Y.; and Tennenholtz, M. 2010. Transferable Utility Planning Games. In *Procs AAAI 2010*. AAAI Press.

Cox, J. S., and Durfee, E. H. 2005. An efficient algorithm for multiagent plan coordination. In *Procs AAMAS 2005*, 828–835. ACM.

Cox, J. S.; Durfee, E. H.; and Bartold, T. 2005. A distributed framework for solving the multiagent plan coordination problem. In *Procs AAMAS 2005*, 821–827. ACM.

Dimopoulos, Y., and Moraitis, P. 2006. Multi-agent Coordination and Cooperation through Classical Planning. In *Procs IAT 2006*, 398–402. IEEE Computer Society.

Durfee, E., and Lesser, V. 1991. Partial global planning: A coordination framework for distributed hypothesis formation. *IEEE Transactions on Systems, Man, and Cybernetics* SMC-21(5): 1167–1183.

Durfee, E. H.; Lesser, V. R.; and Corkill, D. D. 1985. Increasing coherence in a distributed problem-solving network. In *Procs IJCAI 1985*, 1025–1030.

Durfee, E. H. 1999. Distributed Problem Solving and Planning. In Weiß, G., ed., *Multiagent Systems*, 121–164.

Ephrati, E., and Rosenschein, J. S. 1994. Multi-agent planning as search for a consensus that maximizes social welfare. In Castelfranchi, C., and Werner, E., eds., *Artificial Social Systems*, number 830 in Lecture Notes in Artificial Intelligence, 207–226. Springer-Verlag.

Grosz, B. J., and Kraus, S. 1996. Collaborative plans for complex group action. *Artificial Intelligence* 86(2): 269–357.

Haslum, P. 2008. Additive and reversed relaxed reachability heuristics revisited. *IPC 2008 Competition Booklet*, ICAPS-08.

Larbi, R. B.; Konieczny, S.; and Marquis, P. 2007. Extending Classical Planning to the Multi-agent Case: A Game-Theoretic Approach. In *Procs ECSQARU'07*, volume 4724 of *Lecture Notes in Artificial Intelligence*, 731–742. Springer-Verlag.

Monderer, D., and Shapley, L. S. 1996. Potential Games. *Games and Economic Behaviour* 14(1): 124–143.

Nissim, R.; Brafman, R. I.; and Domshlak, C. 2010. A general, fully distributed multi-agent planning algorithm. In *Procs AAMAS 2010*, 1323–1330. IFAAMAS.

Richter, S., and Westphal, M. 2010. The LAMA Planner: Guiding Cost-Based Anytime Planning with Landmarks. *Journal of Artificial Intelligence Research* 39: 127–177.

Rosenthal, R. W. 1973. A class of games possessing pure-strategy Nash equilibria. *International Journal of Game Theory* 2: 65–67.

Witteveen, C.; Roos, N.; van der Krogt, R.; and de Weerdt, M. 2005. Diagnosis of single and multi-agent plans. In *Procs AAMAS 2005*, 805–812. ACM.