

# Planning Games

**Ronen I. Brafman**  
Computer Science Dept.  
Ben-Gurion Univ., Israel  
brafman@cs.bgu.ac.il

**Carmel Domshlak and Yagil Engel**  
Industrial Eng. and Management  
Technion–Israel Inst. of Technology  
dcarmel,yagile@ie.technion.ac.il

**Moshe Tennenholtz**  
Microsoft Israel R&D Center  
& Technion–IIT  
moshet@microsoft.com

## Abstract

We introduce planning games, a study of interactions of self-motivated agents in automated planning settings. Planning games extend STRIPS-like models of single-agent planning to systems of multiple self-interested agents, providing a rich class of structured games that capture subtle forms of local interactions. We consider two basic models of planning games and adapt game-theoretic solution concepts to these models. In both models, agents may need to cooperate in order to achieve their goals, but are assumed to do so only in order to increase their net benefit. For each model we study the computational problem of finding a stable solution and provide efficient algorithms for systems exhibiting acyclic interaction structure.

## 1 Introduction

Work in domain-independent AI planning has made substantial progress in recent years — today, we can efficiently solve sophisticated problems of varying expressivity, and we better understand the relationship between the problem structure and the worst-case time complexity of solving it. Much progress has been made in the area of multi-agent (MA) systems as well: great performance improvements have been achieved in various concrete domains, and computational game theory became an important theoretical basis for MA systems research. Still, multi-agent planning seems to lack simple generic models that extend the basic single-agent STRIPS planning model (and its various off-springs) with appropriate game-theoretic constructs.

An obvious difference between single-agent planning and multi-agent planning is that each agent brings into a MA system its own abilities, and together the agents are able to achieve more than they could achieve in isolation. Even the basic setting of a fully cooperative MA system, where the goal definition remains similar to that of single-agent planning, raises interesting computational questions, some of which have been explored within the scope of the multi-entity model [Moses and Tennenholtz, 1995] and MA-STRIPS planning [Brafman and Domshlak, 2008]. More so, when agents are *self-interested*, immediate and well known complications arise, and additional structure is needed.

The first issue is modeling the personal *costs* and *benefits* of each agent. The cost is naturally captured by the total cost of the actions carried out by the agent. The benefit component is less obvious: the agents may either attempt to achieve a joint goal that has some (possibly different) value to each agent, or they may pursue individual goals. The second issue is that of the solution concept. In contrast to MA planning for fully cooperative agents, the solution concept is no longer obvious once the agents are self-interested. For instance, if agent  $\varphi$  prefers plan  $\pi$  to plan  $\pi'$ , while agent  $\varphi'$  prefers plan  $\pi'$  to plan  $\pi$ , which plan should be chosen for the system? What should be criteria for selecting one solution among a set of possible solutions is the basic question tackled in game-theory; this suggests incorporating some game-theoretic ideas in planning models.

Previous work have considered planning for self-interested agents in fully adversarial settings (e.g., [Ben Larbi *et al.*, 2007; Bowling *et al.*, 2003]). In contrast, we consider MA planning for self-interested, yet ready to cooperate agents. We approach this problem both conceptually and computationally. The challenge here is to define natural classes of strategic games and respective solution concepts that properly capture the context of MA planning. We suggest two basic models corresponding to variants of what we call *planning games*. The precise strategy space vary between the two models, but in both, each strategy is a course of action of a single agent (referred as *local plan*), and a joint strategy is a multi-agent plan of the whole system.

In our first model, called *coalition-planning games*, each agent has its own goal, as well as a set of actions it is able to perform. To achieve its goal, an agent may either need, or just find it cost-efficient, to be assisted by some other agents. The question is, when would a coalition of agents work jointly to achieve their individual goals without being tempted to do something else. In other words, under what conditions a plan for such a MA system will be *stable*? We consider a coalition's joint plan to be stable if no subset of its agents would benefit by joining an alternative coalition. Capturing the specifics of planning agents, this concept is in the spirit both of the strong equilibrium in non-cooperative games [Aumann, 1959], and of the core in Non-Transferable-Utility (NTU) cooperative games [Aumann and Peleg, 1960].

Our second model, called *auction-planning games*, captures problems in which coalitions of agents compete for the

achievement of a single goal which yields a monetary reward. For instance, consider an auction in which coalitions bid for a construction contract. Such a coalition must not only be able to achieve the goal, but also have a winning bid. Moreover, its members should have no reason to switch to another coalition. Capturing all that, our concept of stable solutions is in the spirit of the classical core [von Neumann and Morgenstern, 1944], yet adapted to the strategic setting of self-interested planning agents.

Planning games address many realistic problems in which the complexity of the domain stems from the interplay of game-theoretic and planning issues. However, planning games also contributes to computational game-theory; by introducing new structures of agent interaction that go beyond the previous work on local structure in games (e.g. [Kearns and Littman, 2001; Leyton-Brown and Tennenholtz, 2003; Jeong and Shoham, 2005]). In graphical games, for instance, the payoff of each agent depends on a small number of neighboring agents [Kearns and Littman, 2001]. In contrast, in planning games each agent can depend indirectly on every other agent, because the outcome of a joint plan depends on all agents participating in it. Despite that, for both types of planning games, we show that when a certain graphical structure induced by the system is acyclic, stable plans can be found in time polynomial in the description size of the MA system.

## 2 Background

We assume familiarity with the basic concepts of AI planning, and in particular, the STRIPS formalism. Our formalism and algorithms build upon the work of Brafman and Domshlak [2008] (BD, for short) who investigated the worst-case time complexity of planning for cooperative MA systems. BD introduce a minimal extension of STRIPS to the cooperative MA setting. A MA-STRIPS problem for a system of agents  $\Phi = \{\varphi_i\}_{i=1}^k$  is given by a 4-tuple  $\Pi = \langle P, \{A_i\}_{i=1}^k, I, G \rangle$  where  $P$  is a finite set of atoms,  $I \subseteq P$  and  $G \subseteq P$  encode the initial state and goal, respectively, and, for  $1 \leq i \leq k$ ,  $A_i$  is the set of actions that agent  $\varphi_i$  is capable of performing. Each action  $a \in A = \bigcup A_i$  has the STRIPS syntax and semantics;  $a = \langle \text{pre}(a), \text{add}(a), \text{del}(a) \rangle$  is given by its preconditions, add effects, and delete effects.

BD make the *simple agents* assumption that the (suitably formalized) local planning problem of each agent can be solved in polynomial time. In this case, the sole cause of complexity of MA planning stems from the interactions between agents. Intuitively, if the agents are completely independent, then MA planning reduces to multiple independent single-agent planning problems, and consequently, the overall time complexity is linear in the number of agents. However, if the MA system is strongly coupled, the complexity might be exponential in the number of agents. BD identify a concrete, quantifiable parameter of the MA system that captures its degree of coupling, and show that any MA-STRIPS problem can be compiled into a constraint satisfaction problem (CSP), efficiently solvable when that parameter’s value is not too high.

We adopt and, for ease of presentation further strengthen the *simple agents* assumption, requiring each agent’s plan to

be bounded in length. This simplifies some technicalities, allowing us to focus on the game-theoretic constructions; all our results extend with no complications to simple agents with unbounded local plans. Moreover, the key computational questions remain—the size of the joint plans grows linearly, and thus the size of the plan space grows exponentially with the number of agents.

We highlight the following technical details of BD’s compilation of a planning problem  $\Pi$  into a CSP.

- (a) The CSP of  $\Pi$  is defined over variables bijectively corresponding to the agents  $\Phi$ , with the variable domain  $\mathcal{D}(\varphi)$  containing sequences of annotated actions, bounded in length by a fixed  $\delta$ .
- (b) Each annotated action of agent  $\varphi_i$  is a tuple  $(a, t, \{(j_1, t_1), \dots, (j_m, t_m)\})$  where  $a \in A_i$ ,  $t$  and all  $t_i$  are time points, and each  $j_i$  is an identity of some other agent. The semantics of such a tuple is “at time  $t$ ,  $\varphi_i$  will perform action  $a$ , and it requires agents  $\varphi_{j_i}$  to provide the  $j_i$ -th precondition of  $a$  at time  $t_{j_i}$ .”
- (c) There are unary and binary constraints. Each unary constraint ensures that a sequence of annotated actions is self-consistent. That is, if one annotated action of  $\varphi$  requires it to supply itself with some precondition, then another annotated action in the sequence must produce this precondition at the right time. Such self-consistent sequences of annotated actions are called the agent’s *strategies*. Strategies may include less than  $\delta$  actions; in particular, we use  $\perp$  (the *null strategy*) to denote an empty sequence. The binary constraints capture inter-agent dependencies.  $\theta \in \mathcal{D}(\varphi)$  and  $\theta' \in \mathcal{D}(\varphi')$  satisfy their constraint if whenever  $\theta'$  requires a precondition by  $\varphi$  at time  $t$ ,  $\theta$  contains an action producing this precondition at  $t$ , and vice versa. In that case, we say that  $\theta$  and  $\theta'$  *match*.
- (d) The constraint network induced by that CSP is isomorphic to the *agent interaction graph* (AIG)  $IG_\Pi$  in which the nodes are the agents, and there is an edge between two agents if an action of one of them either adds or deletes a precondition of some action of the other agent.

Any set of pair-wise matching strategies  $\theta_1, \dots, \theta_k$  of the respective agents (that is, any solution to the CSP), constitutes a valid plan for  $\Pi$ . BD show that such a CSP can be solved in time exponential only in  $w\delta$ , where  $w$  is the *tree-width* of  $IG_\Pi$ . In particular, if  $IG_\Pi$  is acyclic, then  $w = 1$ , and thus  $\Pi$  can be solved in time exponential only in  $\delta$ , regardless of the number of agents involved.

## 3 Coalition-Planning Games

BD exploit the structure of the MA system to achieve tractability, within a model that assumes that agents are fully cooperative, working together to achieve a *mutual goal*. In many realistic settings, however, agents are self-interested, have personal goals and costs, and are motivated to act to increase their personal net benefit. Yet, because different agents may have different capabilities, they may still find it beneficial to cooperate with each other. This creates some interesting tension between their private utility maximization and the need to provide incentives to other agents whose help they

desire. To capture such settings, which introduce a game-theoretic flavor into the planning problem, we consider a minimal extension of the MA-STRIPS formalism to self-interested agents, which we call *coalition-planning games (CoPG)*. MA-STRIPS extended STRIPS by associating actions with agents, and CoPG extends MA-STRIPS by associating a single subgoal with each agent and associating personal costs with actions. The subgoals are associated with personal values, or reward, as well. Formally:

**Definition 1** A *coalition-planning game (CoPG)* for a system of agents  $\Phi$  is a tuple  $\Pi = \langle P, A, I, G, c, r \rangle$  where atoms  $P$ , initial state  $I$ , and actions  $A$  are defined as in a MA-STRIPS.  $G \subset P$  is a set of goal atoms, one per agent, with  $g_\varphi \in G$  denoting the personal goal of agent  $\varphi$ ,  $c$  is an action cost function  $c : A \rightarrow \mathbb{R}^+$ , and  $r : \Phi \rightarrow \mathbb{R}$  is a function capturing the reward each agent associates with its own goal.

A joint strategy of the agents induces a (possibly parallel) plan  $\pi$  that contains the actions each agent performs at each time step according to its own strategy. We use  $\pi|_\varphi$  to denote the actions of agent  $\varphi$  within  $\pi$ . The reward agent  $\varphi$  obtains from plan  $\pi$ , denoted by  $r_\varphi(\pi)$ , is  $r(\varphi)$  if  $\pi$  achieves  $g_\varphi$ , and zero otherwise. The personal cost of each agent in a plan  $\pi$ , denoted  $c_\varphi(\pi)$ , is the sum of costs of its actions in  $\pi$ , i.e.,  $\sum_{a \in \pi|_\varphi} c(a)$ . As  $\pi|_\varphi$  corresponds to a strategy  $\theta \in \mathcal{D}(\varphi)$ , we also use  $c_\varphi(\cdot)$  for strategies, as in  $c_\varphi(\theta)$ . The net value of a plan  $\pi$  to agent  $\varphi$  (or its *utility*) is  $u_\varphi(\pi) = r_\varphi(\pi) - c_\varphi(\pi)$ .

As common in situations where cooperation of selfish agents is required, we look for solutions which have some notion of stability. Intuitively, a solution is *stable* if there exists no set of agents, all of which can increase their utility by jointly adopting a different plan.

**Definition 2** A solution  $\pi$  for a coalition-planning game  $\Pi$  of agents  $\Phi$  is called **stable** iff there is no alternative plan  $\pi'$  involving a subset of agents  $\Phi' \subseteq \Phi$  such that  $u_\varphi(\pi') > u_\varphi(\pi)$  for all  $\varphi \in \Phi'$ .

Roughly speaking, our definition of stability is in the spirit of strong equilibrium in non-cooperative games. A strong equilibrium [Aumann, 1959] is a strategy profile, one strategy for each agent, such that a deviation by any subset of the agents will not be beneficial to at least one of them, assuming that the other agents stick to their strategies. In a planning domain the idea of “sticking” to strategies may be problematic, since these may require pre-conditions that the deviating agents no longer supply. In our definition of stability we therefore assume that the other agents take the “null action”, i.e., do not interfere with the deviation. This is consistent with the idea that a stable plan is one in which no subset of agents can “complain” that they can all do better. In the full version of this paper we also consider the case where the other agents are adversarial, i.e., they may perform arbitrary actions that could interfere with the deviating plan. The latter is similar to the notion of  $c$ -acceptable strategies defined by Aumann [1959]. These definitions are also in the spirit of the NTU-core [Aumann and Pelleg, 1960], but this concept refers to coalition outcomes rather than to strategic agent behavior.

**algorithm** CoPG-Acyclic( $\Pi, \Phi$ )

input: an acyclic CoPG  $\Pi$  over  $k$  agents  $\Phi$

output: a stable plan for  $\Pi$

fix a topological ordering  $\varphi_1, \dots, \varphi_k$  over  $\Phi$

for  $i = k$  down to 1:

  for each  $\theta \in \mathcal{D}(\varphi_i)$ :

    if for some child  $\varphi_j$  of  $\varphi_i$ ,  $\theta$  has no matches in  $\mathcal{D}(\varphi_j)$  then  
      remove  $\theta$  from  $\mathcal{D}(\varphi_i)$

$\mathcal{D}^*(\varphi_i) = \{\theta \in \mathcal{D}(\varphi_i) \mid$

$\theta$  does not require preconditions from  $\varphi_i$ 's parent}

$\hat{u}_\varphi = \max_{\theta \in \mathcal{D}^*(\varphi)} \bar{u}_\varphi(\theta)$

  for each  $\theta \in \mathcal{D}(\varphi_i)$ :

    if  $\bar{u}_{\varphi_i}(\theta) < \hat{u}_{\varphi_i}$  then remove  $\theta$  from  $\mathcal{D}(\varphi_i)$

for  $i = 1$  to  $k$ :

  select  $\theta_{\varphi_i} \in \mathcal{D}(\varphi_i)$  matching  $\theta_{\varphi_j}$  selected for the parent  $\varphi_j$  of  $\varphi_i$

  merge  $\{\theta_{\varphi_1}, \dots, \theta_{\varphi_k}\}$  into a joint plan  $\pi$

return  $\pi$

Figure 1: Algorithm for stable planning in acyclic coalition-planning games.

We now show that under the “simple agent” assumption discussed earlier, planning for coalition-planning games forming *acyclic* agent interaction graph is tractable. Our algorithm is based on the common message-passing approach for solving graphical reasoning problems such as CSPs. However, as stability does not seem to have a direct CSP interpretation, we provide a special purpose algorithm which we describe below. To simplify the presentation, we pose the (easy to drop) assumption that the goals of all the agents are different, and  $\varphi$  will be the only agent that can achieve  $g_\varphi$ .

The overall flow of the algorithm is similar to the two-pass algorithm for solving CSP with an acyclic constraint graph, which is conveniently viewed as a tree. At the first, bottom-up phase, each node  $\varphi$  in its turn, gets a message with  $\mathcal{D}(\varphi')$  from each of its children  $\varphi'$ , all of whom were previously processed. Given these children’s domains,  $\varphi$  removes from  $\mathcal{D}(\varphi)$  any strategy that does not match at least one strategy in the domain of each child. At the second, top-down phase, we select an arbitrary strategy  $\theta_\varphi$  of the root variable  $\varphi$ , and pass it to  $\varphi$ ’s children. Given  $\theta_\varphi$ , each child  $\varphi'$  of  $\varphi$  selects a strategy  $\theta_{\varphi'} \in \mathcal{D}(\varphi')$  that matches  $\theta_\varphi$ , and passes it to its own children. Such a matching strategy  $\theta_{\varphi'}$  is guaranteed to exist, or otherwise  $\theta_\varphi$  would have been discarded at the bottom-up phase. The process proceeds this way down the tree until it reaches the leaves, and a valid plan is then straightforwardly constructed from the selected strategies  $\{\theta_{\varphi_i}\}_{i=1}^k$ .

By itself, the above scheme finds a plan, but not necessarily a stable one. In order to limit the algorithm’s output to stable plans, we add a stability check to the bottom-up phase, as follows. First, we define the *potential utility*  $\bar{u}_\varphi(\theta_\varphi)$  of agent  $\varphi$  from its strategy  $\theta_\varphi$  as  $\bar{u}_\varphi(\theta_\varphi) = r(\varphi) - c_\varphi(\theta_\varphi)$ . It is “potential” because it is realized only if  $\theta_\varphi$  is completed to a valid plan  $\pi$ , in which case  $u_\varphi(\pi) = \bar{u}_\varphi(\theta_\varphi)$ .

Let  $\mathcal{D}^*(\varphi) \subseteq \mathcal{D}(\varphi)$  denote the set of strategies in the pruned domain of  $\varphi$  that do not require the support of  $\varphi$ ’s parent. That is, all the requests (if any) to support preconditions in a strategy  $\theta_\varphi \in \mathcal{D}^*(\varphi)$  are only from the children of  $\varphi$ . Let  $\hat{u}_\varphi = \max_{\theta \in \mathcal{D}^*(\varphi)} \bar{u}_\varphi(\theta)$ . That is,  $\hat{u}_\varphi$  is the highest

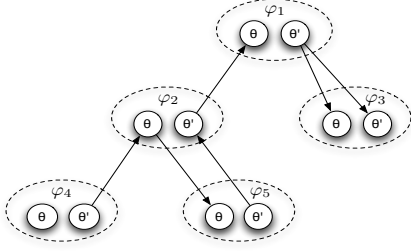


Figure 2: Agent interaction graph for the example problem. Each node is annotated with the strategy set of the respective agent. An arrow from strategy  $\theta_\varphi$  to strategy  $\theta_{\varphi'}$  indicates that  $\theta_\varphi$  provides precondition required by  $\theta_{\varphi'}$  from  $\varphi$ .

utility  $\varphi$  can get from a plan that does not involve any of its ancestors. We discard from  $\mathcal{D}(\varphi)$  all strategies  $\theta$  such that  $\bar{u}_\varphi(\theta) < \hat{u}_\varphi$ . Note that in particular any strategy  $\theta$  whose induced cost  $c_\varphi(\theta)$  is greater than  $r(\varphi)$  is discarded, and if  $\mathcal{D}^*(\varphi)$  includes any strategy with induced cost lower than  $r(\varphi)$ , then  $\perp$  is also discarded from  $\mathcal{D}(\varphi)$ .

Figure 1 summarizes the algorithm, and we now provide a small example illustrating it. Consider a coalition-planning game over five agents  $\Phi = \{\varphi_1, \dots, \varphi_5\}$ , with the agent interaction graph being as depicted in Figure 2. Assume that each agent  $\varphi$  has two possible goal-achieving strategies,  $\theta_\varphi$  and  $\theta'_\varphi$ , with induced costs  $c_\varphi(\theta_\varphi) < c_\varphi(\theta'_\varphi)$ , and that the personal utilities  $r(\varphi)$  exceed these costs, providing  $\bar{u}_\varphi(\theta_\varphi) > \bar{u}_\varphi(\theta'_\varphi) > 0$ . The inter-agent relationship between the personal agent strategies is depicted in Figure 2; for instance,  $\theta_{\varphi_1}$  requires some preconditions from  $\varphi_2$ , and  $\varphi_2$  can provide the required support by adopting  $\theta'_{\varphi_2}$ .

In the bottom-up phase,  $\mathcal{D}^*(\varphi_4) = \{\theta_{\varphi_4}, \theta'_{\varphi_4}, \perp\}$ , therefore  $\hat{u}_{\varphi_4} = \bar{u}_{\varphi_4}(\theta_{\varphi_4})$ .  $\theta'_{\varphi_4}$  and  $\perp$  provide lower utility to  $\varphi_4$ , and thus are discarded, and the refined domain  $\mathcal{D}(\varphi_4) = \{\theta_{\varphi_4}\}$  is sent to  $\varphi_2$ .  $\varphi_5$  has  $\mathcal{D}^*(\varphi_5) = \{\theta'_{\varphi_5}, \perp\}$ , hence  $\hat{u}_{\varphi_5} = \bar{u}_{\varphi_5}(\theta'_{\varphi_5})$ , and only  $\perp$  is discarded from  $\mathcal{D}(\varphi_5)$ . Next,  $\varphi_2$  finds that  $\theta_{\varphi_4}$  (the only strategy received from  $\varphi_4$ ) matches its strategies  $\theta'_{\varphi_2}$  and  $\perp$ , but not  $\theta_{\varphi_2}$  (because  $\theta_{\varphi_4}$  does not provide the requirements of  $\theta_{\varphi_2}$  from  $\varphi_4$ ). Therefore, the strategy  $\theta_{\varphi_2}$  is discarded from  $\mathcal{D}(\varphi_2)$ , and the modified domain  $\{\theta'_{\varphi_2}, \perp\}$  is now evaluated against  $\mathcal{D}(\varphi_5)$ , with  $\theta'_{\varphi_5}$  matching both  $\theta'_{\varphi_2}$  and  $\perp$ . Here, too,  $\perp$  is discarded as  $\varphi_2$  has proved to have a better alternative  $\theta'_{\varphi_2}$ . Proceeding now with agent  $\varphi_3$ , both its non-null strategies require  $\varphi_1$ 's support, and thus  $\mathcal{D}^*(\varphi_3) = \{\perp\}$ ,  $\hat{u}_{\varphi_3} = 0$ , and no strategy is discarded from the domain. When  $\varphi_1$  is processed,  $\theta_{\varphi_1}$  matches only  $\perp$  in  $\mathcal{D}(\varphi_3)$ , whereas  $\theta'_{\varphi_1}$  matches all strategies of  $\mathcal{D}(\varphi_3)$ . Both strategies also match  $\theta'_{\varphi_2}$ , and  $\theta'_{\varphi_1}$  also matches  $\perp$  of  $\varphi_2$ . As both strategies of  $\varphi_1$  have a match with each child,  $\hat{u}_{\varphi_1} = \bar{u}_{\varphi_1}(\theta_{\varphi_1})$  and  $\theta'_{\varphi_1}$  and  $\perp$  are discarded. Table 1 shows the domains of the nodes after processing. For the top-down phase, the only possible joint strategy is  $(\theta_{\varphi_1}, \theta'_{\varphi_2}, \theta_{\varphi_4}, \theta'_{\varphi_5})$  and  $\perp$  for  $\varphi_3$ . There are other feasible joint strategies in which all agents achieve their goals, for example  $(\theta'_{\varphi_1}, \theta'_{\varphi_2}, \theta'_{\varphi_4}, \theta'_{\varphi_5}, \theta_{\varphi_3})$ . However, the latter joint strategy has several deviations. For example,  $\varphi_1$  deviates to

$\varphi$	$\mathcal{D}^*(\varphi)$	$\hat{u}_\varphi$	$\mathcal{D}(\varphi)$
$\varphi_4$	$\{\theta_{\varphi_4}, \theta'_{\varphi_4}, \perp\}$	$\bar{u}_{\varphi_4}(\theta_{\varphi_4})$	$\{\theta_{\varphi_4}\}$
$\varphi_5$	$\{\theta'_{\varphi_5}, \perp\}$	$\bar{u}_{\varphi_5}(\theta'_{\varphi_5})$	$\{\theta_{\varphi_5}, \theta'_{\varphi_5}\}$
$\varphi_2$	$\{\theta'_{\varphi_2}, \perp\}$	$\bar{u}_{\varphi_2}(\theta'_{\varphi_2})$	$\{\theta'_{\varphi_2}\}$
$\varphi_3$	$\{\perp\}$	0	$\{\theta_{\varphi_3}, \theta'_{\varphi_3}, \perp\}$
$\varphi_1$	$\{\theta_{\varphi_1}, \theta'_{\varphi_1}, \perp\}$	$\bar{u}_{\varphi_1}(\theta_{\varphi_1})$	$\{\theta_{\varphi_1}\}$

Table 1: The result of the bottom-up processing for the example; the domains  $\mathcal{D}(\varphi)$  are shown after processing.

$\theta_{\varphi_1}$ , and  $\{\varphi_2, \varphi_5\}$  jointly deviate to their  $\theta$  strategies.

**Theorem 1** For any coalition-planning game  $\Pi$  over simple agents, if  $IG_\Pi$  is acyclic, then a stable plan exists, and algorithm *CoPG-Acyclic* returns such a plan.

Note that the algorithm does not return all stable plans, but it is guaranteed to return at least one. Also, if the null plan ( $\perp$  strategy for all agents) is stable, then it is the only stable plan.

## 4 Auction-Planning Games

In coalition-planning games agents are motivated by their selfish goals but do not compete with each other. We now consider scenarios in which each agent aims at joining a coalition that jointly satisfies a goal, and does so in cost lower than competing coalitions. Satisfying a goal yields a certain monetary gain on whose distribution the related coalition must decide. Perhaps most immediate application of this model is procurement auctions. While classical auction theory considers the bidding agents and their values to be fixed, in practice bidding may require cooperation of multiple self-ish parties, each with its own capabilities (e.g., products it can supply under different preconditions with different costs and constraints). Our model captures this process of preparation for bidding where agents must combine their capabilities in a non-trivial manner.

First, we consider a setting in which the winning coalition gets a fixed bonus plus its plan's cost. We then consider a setting in the spirit of 2nd price (reverse) auction; here the winning coalition is paid the cost associated with the cheapest plan of the complementary coalition. In both cases (i) the gains are to be distributed among the coalition members, and (ii) a coalition is stable if no subset of its members can join some non-winning agents and form a joint plan with a distribution of payments that is more beneficial to all. This setting induces a game with side payments, and the solution concept is in the spirit of the core solution. However, in our setting, the value of a coalition is determined by its possible joint plans and strategic distribution of gain, rather than being given exogenously. In addition, our solution concept considers deviations by coalitions involving both winning and non-winning members. Hence, planning games suggest new, natural ways of combining elements of cooperative and non-cooperative game theory.

**Definition 3** An auction-planning game (AuPG) for a system of agents  $\Phi$  is a tuple  $\Pi = \langle P, A, I, G, c, r \rangle$ , where  $P$ ,  $A$ ,  $I$ , and  $G$  are as in MA-STRIPS,  $c$  is an action cost function

$c : A \rightarrow \mathbb{R}^+$ , and  $r : A^* \rightarrow \mathbb{R}^+$  is a specification of the reward of the coalition selected to achieve the goal.

A solution  $\sigma$  for a AuPG is a triplet  $\langle \pi_\sigma, \Gamma_\sigma, \hat{r}_\sigma \rangle$  where  $\pi_\sigma$  is a MA-STRIPS plan for  $\Pi$ ,  $\Gamma_\sigma \subseteq \Phi$  is a coalition performing  $\pi_\sigma$ , and  $\hat{r}_\sigma : \Gamma_\sigma \rightarrow \mathbb{R}^+$  is a division of the reward  $r(\pi_\sigma)$  among the coalition members such that  $\sum_{\varphi \in \Gamma_\sigma} \hat{r}_\sigma(\varphi) = r(\pi_\sigma)$ . We make a natural assumption that  $r(\cdot)$  is monotonically increasing in the cost of the chosen coalition. Hence, given a partition of  $\Phi$  into coalitions, the coalition having the lowest-cost plan is selected and granted the reward.

#### 4.1 Fixed-Bonus Reward Model

In our first setting, the winning coalition is granted the cost of its plan  $c(\pi_\sigma) = \sum_{a \in \pi_\sigma} c(a)$  plus some fixed bonus  $B > 0$ , that is,  $r(\pi_\sigma) = c(\pi_\sigma) + B$ . A rational agent will not participate in a plan for a personal reward lower than its personal costs in that plan, and thus each member  $\varphi \in \Gamma_\sigma$  is paid  $c_\varphi(\pi) + b_\sigma(\varphi)$ , such that  $\sum_{\varphi \in \Gamma_\sigma} b_\sigma(\varphi) = B$ , and hence  $\varphi$ 's utility is  $u_\varphi(\sigma) = b_\sigma(\varphi)$ . Because the reward is distributed within the winning coalition, the utility of the agents outside it is trivially zero. With divisible utilities, coalitions will always (weakly) prefer to use their lowest-cost plan. We use  $c^*(\Gamma)$  to denote the lowest cost with which  $\Gamma$  can attain the goal;  $c^*(\Gamma) = \infty$  denotes inability of  $\Gamma$  to attain the goal.

**Definition 4** A solution  $\sigma$  is a **winning bid** if for any  $\Gamma \subseteq \Phi \setminus \Gamma_\sigma$ , and for any  $\Gamma' \subset \Gamma_\sigma$ , holds  $c(\pi_{\sigma'}) < c^*(\Gamma)$ . That is, neither agents outside  $\Gamma_\sigma$ , nor any strict subset of  $\Gamma_\sigma$ , have a plan with a lower cost.

This definition is based on an implicit model of the auction issuer, or auctioneer, who chooses the coalition for which the reward is the lowest. Furthermore, the auctioneer prevents manipulation of the reward by not selecting coalitions which include redundant members, because such coalitions potentially inflate their bonus as those redundant members could otherwise be part of a competing coalition.

**Definition 5** A winning-bid  $\sigma$  is **stable** if there is no winning-bid  $\sigma' \neq \sigma$  with  $u_\varphi(\sigma') > u_\varphi(\sigma)$  for all  $\varphi \in \Gamma_{\sigma'}$ .

We say that a coalition is stable if it has a stable winning-bid solution. It turns out that stable coalitions in fixed-bonus AuPGs have a verifiable topological characterization.

**Lemma 2** A coalition  $\Gamma_\sigma$  with a winning-bid solution  $\sigma$  is stable unless there exist coalitions  $\Gamma$  and  $\Gamma'$ , both having goal achieving plans, such that (i) the three coalitions are pairwise non-disjoint, and (ii) neither  $\Gamma_\sigma \cap \Gamma'$  is included in  $\Gamma_\sigma \cap \Gamma$  nor the other way round. Furthermore, if such  $\Gamma$  and  $\Gamma'$  do exist, and both have winning-bids, then  $\Gamma_\sigma$  is not stable.

Lemma 2 implies that refuting the stability of a winning-bid requires a cycle of coalitions. Because a winning-bid coalition is required to be minimal, and thus connected in the agent interaction graph, a cycle of coalitions implies a cycle in the AIG. Thus, if the AIG is acyclic, any winning-bid is stable. We can therefore construct a stable solution  $\sigma$  as follows: (i) find a cost-optimal plan  $\pi$  for  $\Phi$ , (ii) define  $\Gamma_\sigma$  to

be the set of agents that participate in  $\pi$ , and (iii) allocate the reward to prevent deviations by intersecting subsets. The last step can be done efficiently similar to the method used by the algorithm we present below.

#### 4.2 Second-Cost Reward Model

We now take the auction analogy one step further, and define the reward to be the cost of the second-best solution. This model strengthens the attractiveness of lower-cost coalitions because they have higher utility to distribute among their agents. The reward specification includes a reserve price  $\rho$ , denoting the maximal possible reward, that is,  $r(\pi_\sigma) = \min\{\rho, \min_{\Gamma \subseteq \Phi \setminus \Gamma_\sigma} c^*(\Gamma)\}$ .

As with a fixed bonus, acyclic agent-interaction graphs are easier to handle because for a given coalition there exists a single threat on its stability. Therefore, a coalition that obtains the best possible bonus (the *bonus-optimal* coalition), must be stable. Note that the bonus-optimal plan may not be the cost-optimal one; a higher cost coalition may have weaker competition by its complement and hence achieve a higher bonus. In order to find a bonus-optimal coalition, we iterate over the *cuts* of AIG. Each edge  $(\varphi', \varphi'')$  defines a pair of “directional” cuts  $\varphi' \text{---} \varphi''$  and  $\varphi'' \text{---} \varphi'$ . By  $\Gamma_{\varphi' \text{---} \varphi''}$  we denote the nodes on the  $\varphi'$  side of the edge. In addition,  $\pi^*(\Gamma)$  refers to a cost-optimal plan for each  $\Gamma \subseteq \Phi$ . A plan may actively involve only a subset of  $\Gamma$ , denoted henceforth by  $\bar{\Gamma}(\pi^*(\Gamma))$ .

The *cut-bonus-optimal* coalition of  $\varphi' \text{---} \varphi''$  is defined as a bonus-optimal coalition among all coalitions  $\Gamma$  in  $\Gamma_{\varphi' \text{---} \varphi''}$  that fulfill two conditions: (1) they include  $\varphi'$ , (2) their second-best coalition is on the other side of the edge (that is,  $\bar{\Gamma}(\pi^*(\Phi \setminus \Gamma)) \subseteq \Gamma_{\varphi'' \text{---} \varphi'}$ ). The *cut value*  $V(\varphi' \text{---} \varphi'')$  is the bonus obtained by the respective cut-bonus-optimal coalition.

For example, consider the AIG in Figure 3(a), for which the possible goal-achieving coalitions are listed in Figure 3(b) along with their respective costs. The cut-bonus-optimal coalition of the cut  $\varphi_5 \text{---} \varphi_7$  is  $\{\varphi_1, \varphi_4, \varphi_5\}$ , whose second-best coalition is  $\{\varphi_6, \varphi_7\}$  and hence it achieves a bonus of  $7 - 4 = 3$ . The cut-bonus-optimal coalition of the cut  $\varphi_7 \text{---} \varphi_5$  is  $\{\varphi_6, \varphi_7\}$ , a coalition which in fact does not have a winning-bid (or, in other words, obtains negative bonus).

**Lemma 3** Let  $B^*$  be the highest bonus possible for  $\Pi$ . Then there exists a cut  $\varphi' \text{---} \varphi''$  such that  $V(\varphi' \text{---} \varphi'') = B^*$ .

An immediate corollary is that a coalition that obtains a bonus which is the maximum cut value in  $IG_\Pi$ , is a bonus-optimal coalition for  $\Pi$ . It is therefore sufficient to find all the cut values, while recording which coalition achieves each one. Our algorithm, depicted in Figure 4, computes a cut-bonus-optimal coalition, for a cut  $\varphi' \text{---} \varphi''$ , as follows. It first computes a cost-optimal plan, constrained to include  $\varphi'$ .<sup>1</sup> If its second-best is in the other side of the cut, we are done. Otherwise, as shown in the proof of Theorem 4, the path between  $\varphi'$  and that second-best (denoted  $path(\cdot, \cdot)$ ), must be part of the cut-bonus-optimal plan. Hence in each iteration, the set which constrains the cut-bonus-optimal planning problem grows, until the cost-optimal plan converges to

<sup>1</sup>Participation constraint is added by removing  $\perp$  from the domain of the agents that must participate.

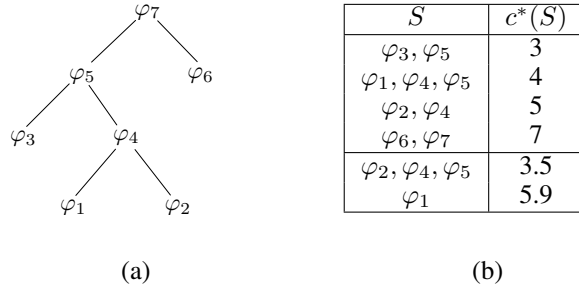


Figure 3: (a) AIG and (b) goal-achieving coalitions and their respective costs of plans, for illustration of AuPG-Acyclic.

**algorithm AuPG-Acyclic**( $\Pi, \Phi$ )

input: an acyclic, second-cost AuPG  $\Pi$  over agents  $\Phi$

output: a stable winning-bid plan for  $\Pi$

for each cut  $\varphi' - \varphi''$ :

loop

$\Omega = \{\varphi'\}$

find  $\Gamma_{\varphi'}^* = \bar{\Gamma}(\pi^*(\Gamma_{\varphi' - \varphi''}))$  under constraint  $\Omega \subseteq \Gamma_{\varphi'}^*$

find  $\hat{\Gamma}_{\varphi'} = \bar{\Gamma}(\pi^*(\Phi \setminus \Gamma_{\varphi'}^*))$

if  $\hat{\Gamma}_{\varphi'} \subseteq \Gamma_{\varphi' - \varphi''}$  then endloop

else  $\Omega = \Omega \cup \text{path}(\Omega, \hat{\Gamma}_{\varphi'})$

$V(\varphi' - \varphi'') := c^*(\hat{\Gamma}_{\varphi'}) - c^*(\Gamma_{\varphi'}^*)$

select  $\varphi = \arg \max_{\varphi'} \max_{\varphi''} V(\varphi' - \varphi'')$

$\sigma = \langle \pi^*(\Gamma_{\varphi}^*), \Gamma_{\varphi}^*, \text{Comp-Div}(\Gamma_{\varphi}^*) \rangle$

return  $\sigma$

**procedure Comp-Div**( $\Gamma$ )

perform bottom-up scan of  $\Gamma$ , allocate all the reward to the first node whose subtree in  $IG_{\Pi}$  includes a winning bid.

Figure 4: Algorithm for stable winning-bid planning in acyclic AuPGs with the second-cost reward model.

the cut-bonus-optimal one. The algorithm executes a number of cost-optimal planning problems which is in the worst-case quadratic in  $|\Phi|$ , hence it is polynomial under the simple agents assumption and given the results of BD.

**Theorem 4** For any acyclic AuPG  $\Pi$  with the second-cost reward model, there exists a stable winning-bid solution, and algorithm AuPG-Acyclic returns such a solution.

We demonstrate the algorithm using the AIG in Figure 3(a). Assume that the sets  $S$  of agents, listed in the first four rows of Figure 3(b), are the only coalition with goal achieving plans, and their respective costs are  $c^*(S)$ . The bonus-optimal plan is  $\{\varphi_1, \varphi_4, \varphi_5\}$ , with a bonus of 3, given that its second-best coalition is  $\{\varphi_6, \varphi_7\}$ . We show that the computation of  $V(\varphi_5 - \varphi_7)$  identifies the right coalition and its value. First, we find a cost-optimal plan on  $\Gamma_{\varphi_5 - \varphi_7}$ , under the constraint that  $\varphi_5$  participates, and get  $\Gamma_{\varphi_5}^* = \{\varphi_3, \varphi_5\}$ . Next, we find cost-optimal plan on  $\Phi \setminus \{\varphi_3, \varphi_5\}$ , returning  $\hat{\Gamma}_{\varphi_5} = \{\varphi_2, \varphi_4\}$ . Because this second-best is within the same side of the cut, the loop does not terminate. Now comes the key idea of the algorithm:  $\{\varphi_2, \varphi_4\}$  must intersect with the real bonus-optimal plan. Hence we add the path from  $\varphi_5$  to

$\{\varphi_2, \varphi_4\}$ , which consists just of the node  $\varphi_4$ , to  $\Omega$ . Next we find cost-optimal plan under the constraint that  $\varphi_5$  and  $\varphi_4$  must participate. We get  $\{\varphi_1, \varphi_4, \varphi_5\}$ , and its second best plan is by  $\{\varphi_6, \varphi_7\}$ , which is on the other side of the cut, and hence the loop terminates. If we take the additional two sets in Figure 3(b) into account, that second iteration outputs  $\{\varphi_2, \varphi_4, \varphi_5\}$  with cost 3.5, which is not yet the cut-bonus-optimal because its second best is by  $\{\varphi_1\}$  with cost 5.9. We add  $\varphi_1$  to  $\Omega$ , and the next iteration finds  $\{\varphi_1, \varphi_4, \varphi_5\}$ .

To compute reward division, we find cost-optimal plan of each subtree  $\Gamma_{\varphi_1}$ ,  $\Gamma_{\varphi_4}$ , and  $\Gamma_{\varphi_5}$ , and compare to the cost-optimal plan of its complement to find out whether it is a winning-bid. We find that  $\Gamma_{\varphi_1}$  and  $\Gamma_{\varphi_4}$  do not have a winning-bid (because  $\{\varphi_3, \varphi_5\}$  has a cheaper plan), but  $\{\varphi_3, \varphi_5\} \in \Gamma_{\varphi_5}$  does have one. Hence  $\varphi_5$ 's reward is its cost plus  $B^* = 3$ , while  $\varphi_4$  and  $\varphi_1$  get just their cost back.

Finally, assume that  $c^*(\varphi_1) = c^*(\{\varphi_2, \varphi_4\}) = 6.1$ . The cut-bonus-optimal coalition of  $\varphi_5 - \varphi_7$  is still  $\{\varphi_1, \varphi_4, \varphi_5\}$ , with the same value 3. However,  $V(\varphi_5 - \varphi_4) = 3.1$ , obtained by  $\{\varphi_3, \varphi_5\}$ , hence it is now the bonus-optimal plan.

## Acknowledgements

Ronen Brafman and Carmel Domshlak are partially supported by ISF grant 8254320. Ronen Brafman is also supported in part by the Paul Ivanier Center for Robotics Research and Production Management, and the Lynn and William Frankel Center for Computer Science. Yagil Engel is supported in part by an Aly Kaufman fellowship at the Technion.

## References

- [Aumann and Peleg, 1960] R. J. Aumann and B. Peleg. von Neumann-Morgenstern solutions to cooperative games without side payments. *Bulletin of the American Mathematical Society*, 66:173–179, 1960.
- [Aumann, 1959] R. J. Aumann. Acceptable points in general cooperative n-person games. *Annals of Mathematical Studies*, 40:287–324, 1959.
- [Ben Larbi et al., 2007] R. Ben Larbi, S. Konieczny, and P. Marquis. Extending classical planning to the multi-agent case: A game-theoretic approach. In *ECSQARU*, pages 731–742, 2007.
- [Bowling et al., 2003] M. H. Bowling, R. M. Jensen, and M. M. Veloso. A formalization of equilibria for multiagent planning. In *IJCAI*, pages 1460–1462, 2003.
- [Brafman and Domshlak, 2008] R. Brafman and C. Domshlak. From one to many: Planning for loosely coupled multi-agent systems. In *ICAPS*, pages 28–35, 2008.
- [Jeong and Shoham, 2005] S. Jeong and Y. Shoham. Marginal contribution nets: A compact representation scheme for coalitional games. In *ACM-EC*, pages 193–202, 2005.
- [Kearns and Littman, 2001] M. Kearns and M. L. Littman. Graphical models for game theory. In *UAI*, pages 253–260, 2001.
- [Leyton-Brown and Tennenholtz, 2003] K. Leyton-Brown and M. Tennenholtz. Local-effect games. In *UAI*, pages 772–780, 2003.
- [Moses and Tennenholtz, 1995] Y. Moses and M. Tennenholtz. Multi-entity models. *Machine Intelligence*, 14:63–88, 1995.
- [von Neumann and Morgenstern, 1944] J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944.

## A Proofs

**Theorem 1** *For any coalition-planning game  $\Pi$  over simple agents, if  $IG_{\Pi}$  is acyclic, then a stable plan exists, and algorithm CoPG-Acyclic returns such a plan.*

**Proof:** We assume w.l.o.g. that  $IG_{\Pi}$  is connected. First, we show that the algorithm always returns a plan. The top-down phase cannot fail: if a parent  $\varphi$  has some strategy  $\theta$  in its domain, then each of its children must have a matching strategy, or otherwise  $\theta$  would have been pruned. Hence a failure could occur only if the domain of a node becomes empty in the bottom-up phase. A strategy  $\theta \in \mathcal{D}(\varphi)$  is discarded if either (1)  $\theta$  does not match any strategy in the domain of a child, or (2)  $\bar{u}_{\varphi}(\theta) < \hat{u}_{\varphi}$ . Note that, for any node  $\varphi'$ ,  $\mathcal{D}(\varphi')$  must include an action that does not require the support of its parent  $\varphi$ : either  $\perp$ , or a strategy in  $\mathcal{D}^*(\varphi')$  that caused it to discard  $\perp$ . Therefore,  $\perp$  always has a match in any child's domain and it cannot be pruned by condition (1). Furthermore, if condition (2) prunes  $\perp$ , then there must be some strategy in  $\mathcal{D}^*(\psi)$ . Therefore, the bottom-up phase never prunes the domain of a node completely, and thus the algorithm always returns a joint plan. It is left to show that the joint plan  $\pi$  returned by the algorithm is stable. Assume to the contrary that there is a deviation  $\pi'$  from  $\pi$  for some subset  $\Phi' \subseteq \Phi$ . Let  $\varphi$  denote the agent that is the highest, within  $\Phi'$ , in the topological order used by the algorithm. This ensures that  $\theta'_{\varphi}$  does not require the support of the parent of  $\varphi$ , therefore  $\theta'_{\varphi} \in \mathcal{D}^*(\varphi)$ , and thus  $\bar{u}_{\varphi}(\theta'_{\varphi}) \leq \hat{u}_{\varphi}$ . Likewise,  $\pi'$  must be beneficial to  $\varphi$ , hence  $\bar{u}_{\varphi}(\theta_{\varphi}) = u_{\varphi}(\pi) < u_{\varphi}(\pi') = \bar{u}_{\varphi}(\theta'_{\varphi}) \leq \hat{u}_{\varphi}$ . However, in that case  $\theta_{\varphi}$  is pruned from  $\mathcal{D}(\varphi)$ , contradicting the selection of  $\pi$  in the top-down phase.  $\square$

**Lemma 2** *A coalition  $\Gamma_{\sigma}$  with a winning-bid solution  $\sigma$  is stable unless there exist coalitions  $\Gamma$  and  $\Gamma'$ , both having goal achieving plans, such that (i) the three coalitions are pairwise non-disjoint, and (ii) neither  $\Gamma_{\sigma} \cap \Gamma'$  is included in  $\Gamma_{\sigma} \cap \Gamma$  nor the other way round. Furthermore, if such  $\Gamma$  and  $\Gamma'$  do exist, and both have winning-bids, then  $\Gamma_{\sigma}$  is not stable.*

**Proof:** Assume  $\Gamma_{\sigma}$  is not stable. Then there exists an intersecting coalition  $\Gamma$  with a winning-bid solution. (All the coalitions considered in the proof are goal-achieving.) If  $\Gamma$  is the only coalition intersecting  $\Gamma_{\sigma}$ , then  $\Gamma_{\sigma}$  can allocate the whole bonus  $B$  to one agent  $\varphi \in \Gamma_{\sigma} \cap \Gamma$ . Because  $\Gamma$  cannot suggest more to  $\varphi$ ,  $\Gamma_{\sigma}$  is stable. Hence, there has to be yet another coalition  $\Gamma'$  intersecting  $\Gamma_{\sigma}$ . If  $\Gamma \cap \Gamma_{\sigma} \subseteq \Gamma' \cap \Gamma_{\sigma}$ , then  $\Gamma_{\sigma}$  can again allocate  $B$  to an agent in  $\Gamma \cap \Gamma_{\sigma}$ , preventing deviations to both  $\Gamma$  and  $\Gamma'$ . Now assume  $\Gamma \cap \Gamma' = \emptyset$ . Either  $c^*(\Gamma) \leq c^*(\Gamma')$  or  $c^*(\Gamma) > c^*(\Gamma')$ , hence at most one of these coalitions has a winning-bid solution. Assume w.l.o.g. that it is  $\Gamma$ .  $\Gamma_{\sigma}$  again allocates  $B$  to an agent in  $\Gamma \cap \Gamma_{\sigma}$  and prevents deviation. We omit the other direction.  $\square$

**Lemma 3** *Let  $B^*$  be the highest bonus possible for  $\Pi$ . Then there exists a cut  $\varphi' - \varphi''$  such that  $V(\varphi' - \varphi'') = B^*$ .*

**Proof:** Let  $\Gamma^*$  denote a coalition which obtains the optimal bonus  $B^*$ , and let  $\hat{\Gamma}$  denote its second-best coalition. Let  $\varphi$  denote the topologically highest element in  $\Gamma^*$  (note that  $\Gamma^* \subseteq \Gamma_{\varphi}$ ), and let  $\varphi'$  denote the highest element in  $\hat{\Gamma}$ . If  $\varphi'$  is not a descendant of  $\varphi$ , then no node in  $\hat{\Gamma}$  can be a descendant of any node in  $\Gamma_{\varphi}$ , meaning that the cut  $\varphi - \varphi''$ , where  $\varphi''$  is the parent of  $\varphi$ , separates  $\Gamma^*$  and  $\hat{\Gamma}$ . Therefore,  $V(\varphi - \varphi'')$  is at least the bonus  $B^*$  obtained by  $\Gamma^*$ . Due to optimality of  $B^*$ , it must be exactly  $B^*$ . If  $\varphi'$  is a descendant of  $\varphi$ , then let  $\varphi_1$  denote lowest node in  $\Gamma^*$  on the path between  $\varphi$  and  $\varphi'$ . Now the cut  $\varphi'' - \varphi_1$ , where  $\varphi''$  is the child of  $\varphi_1$  on that path, separates  $\Gamma^*$  and  $\hat{\Gamma}$ , hence  $V(\varphi'' - \varphi_1) = B^*$ .  $\square$

**Theorem 4** *For any acyclic AuPG  $\Pi$  with the second-cost reward model, there exists a stable winning-bid solution, and algorithm AuPG-Acyclic returns such a solution.*

**Proof:** Let  $\varphi''$  denote the parent of  $\varphi'$ . Let  $\Gamma$  denote the real cut-bonus-optimal coalition of  $\varphi' - \varphi''$ , and  $\Gamma'$  denote its second-best (hence  $\Gamma' \subseteq \Gamma_{\varphi''}$ ). We first must show that  $V(\varphi' - \varphi'')$  is computed correctly, that is  $\Gamma = \Gamma^*$ , when the algorithm exits the loop. The algorithm first finds the cost-optimal plan of  $\Gamma_{\varphi'}$ , under the constraint that  $\varphi'$  must participate. It then finds the second-best plan  $\hat{\Gamma}_{\varphi'}$ , that is the best plan of the remaining agents. The loop terminates if that plan is within  $\Gamma_{\varphi''}$ , in which case indeed  $\Gamma = \Gamma^*$ . Otherwise,  $\hat{\Gamma}_{\varphi'} \subset \Gamma_{\varphi'}$  (because it cannot contain  $\varphi'$ ). Now, assume for a moment that  $\Gamma \cap \hat{\Gamma}_{\varphi'} = \emptyset$ . Because  $\Gamma'$ , and not  $\hat{\Gamma}_{\varphi'}$ , is the second best of  $\Gamma$ , it must be the case that  $c^*(\hat{\Gamma}_{\varphi'}) \geq c^*(\Gamma')$ . But then  $\Gamma'$  is the second best of  $\Gamma_{\varphi''}$ , meaning  $\Gamma' = \hat{\Gamma}_{\varphi'}$ , and then  $\hat{\Gamma}_{\varphi'} \subseteq \Gamma_{\varphi''}$ . Therefore,  $\Gamma \cap \hat{\Gamma}_{\varphi'} \neq \emptyset$ . Hence the path from  $\varphi'$  to the highest node in  $\hat{\Gamma}_{\varphi'}$  must be in  $\Gamma$ . We can now add this as a constraint and find cost optimal plan again. In each iteration at least one node is added to this set of nodes  $\Omega$ , which must be in the cut-bonus-optimal plan (note that  $\Omega \subseteq \Gamma_{\varphi''}$ , so  $\hat{\Gamma}_{\varphi'}$  must be disjoint from  $\Omega$ ), hence at some point we find a plan  $\Gamma^*_{\varphi'} = \Gamma$ . The value  $V(\varphi' - \varphi'')$  is correct by a symmetric argument. This proves that edge values are computed correctly. From Lemma 3, the coalition selected  $\Gamma^*_{\varphi}$  is indeed bonus-optimal. The node  $\varphi'$  picked to get the full bonus, is the lowest node in  $\Gamma^*_{\varphi}$  whose subtree includes a winning bid  $\Gamma'$ . If there is another winning bid  $\Gamma''$  that intersects with  $\Gamma^*_{\varphi}$ , it must also intersect  $\Gamma'$ .  $\Gamma''$  is not within the subtree of  $\varphi'$ , and  $\Gamma'$  is within the subtree of  $\varphi'$ , therefore  $\Gamma''$  must include  $\varphi'$ . As the bonus obtained by  $\Gamma''$  is not higher than the current bonus of  $\varphi'$ ,  $\varphi'$  will not deviate.  $\square$