Multi-Robot Negotiation: Approximating the Set of Subgame Perfect Equilibria in General-Sum Stochastic Games

Chris Murray CDMURRAY@CS.CMU.EDU
Geoffrey J. Gordon (presenter) GGORDON@CS.CMU.EDU
Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213

In real-world planning problems, we must reason not only about our own goals, but about the goals of other agents with which we may interact. Often these other agents' goals are neither completely aligned with our own nor directly opposed to them. Instead there are opportunities for cooperation: by joining forces, a group of agents can achieve higher utility for all of its members than the members could achieve by themselves. But, in order to cooperate, the agents must negotiate a mutually acceptable plan from among the many possible ones; and, each agent must trust that the others will follow their parts of the deal.

Research from multi-agent planning has often avoided the problem of making sure that all agents have an incentive to follow a proposed joint plan. Game theoretic algorithms, on the other hand, often don't scale to large planning problems. In this paper we attempt to bridge the gap between these two lines of research.

We model the multi-agent planning problem as a general-sum stochastic game, and we present an efficient algorithm for computing and selecting subgame-perfect Nash equilibria in such games. (These equilibria guarantee that every deviation from the plan is deterred by the threat of a suitable punishment, and every threatened punishment is believable.) We demonstrate our algorithm on a two-agent robotic planning problem, and show that it selects a nearly Pareto-dominant equilibrium in which both players achieve substantially more than they could achieve individually. To our knowledge, this is the first multi-agent planning algorithm which can achieve this level of performance in a game of this size.

There are two problems which make game-theoretic planning difficult: finding equilibria can be hard; and having found some equilibria, the agents must agree on one of them or risk *miscoordination*. To address these problems, we present an efficient algorithm for computing some of the subgame-perfect equilibria of a stochastic game, and we describe a negotiation protocol which gives the agents an incentive to agree on (and then follow) a mutually profitable equilibrium. In more detail, we propose the following strategy: first use our planning algorithm to compute equilibria, then reveal some of these equilibria to the other agents, then repeat until no agent wants to reveal more, and finally use our negotiation protocol to decide among the revealed equilibria.

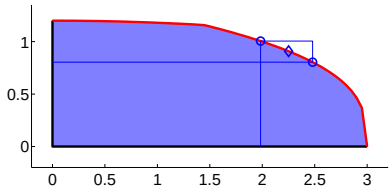

Figure 1. Equilibria of a Rubinstein game with $\gamma = 0.8$. Shaded area shows feasible value vectors $(U_1(x), U_2(x))$ for outcomes x. Right-hand circle corresponds to equilibrium when player 1 moves first, left-hand circle, when player 2 moves first. Nash point is indicated by $\diamond$.

Our planning algorithm performs dynamic programming on a set-based value function: for P players, $V(s) \subset \mathbb{R}^P$ is the set of payoff vectors which we can achieve using some equilibrium policy that we have found so far. We represent $V(s)$ by sampling points on its convex hull. This representation is *conservative*, i.e., guarantees that we find a subset of the true $V^*(s)$. Based on the sampled points we can efficiently compute one-step backups by checking which joint actions are enforceable in an equilibrium.

Our negotiation protocol is based on a multi-player version of Rubinstein's bargaining game, illustrated for two players in Fig. 1. Players take turns proposing divisions of the fruits of cooperation. Each agent can accept the proposal, which fixes her own level of reward and removes her from further play, or can reject and propose a different division of what remains. Until the players agree, the protocol ends with a small probability ϵ after each step and the remaining players get no share of the benefit of cooperation; the fear of

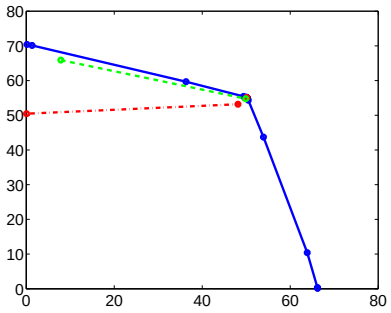

Figure 2. Achieved value compared to approximation accuracy. Solid line: approximate convex hull of value vector set for s_{start}. Dash-dot line: predicted value of negotiated plan for increasing number of samples of hull. Dashed line: actual value in simulation.

this outcome forces players to make reasonable offers. In fact, we can prove that rational agents will immediately offer each other the *Nash bargaining point*, which maximizes the product of their utility gains.

Both our planning and negotiation results depend on all agents knowing a distinguished equilibrium, the *disagreement point*. In theory it is difficult to select a good disagreement point, but in practice the choice is often obvious. For example, in our experiments we used "stay put," which earns $(0, 0)$ in most states.

We tested our value iteration procedure on a simple robot game that lends itself to cooperative strategies. Two players together control a two-wheeled robot, with each player picking the speed of one wheel. Each player wants to cycle through a different list of goals. The world state is $(x, y, \theta, \text{goalindex}_1, \text{goalindex}_2)$; we discretized the problem into 9 joint actions and about 25,000 joint states. Fig. 3 shows an example: each player wants to alternate between opposite corners. The players must choose between the selfish paths which visit only two goals and the cooperative path which visits all four goals and is only slightly longer.

Fig. 2 shows the negotiation problem if both agents have the same computational power, ranging from minutes to hours of wall clock time on a desktop workstation. Fig. 4 shows what happens when we allow the players different amounts of computation. Because the agent with a slower computer doesn't know about some of the best equilibria, the faster agent can influence negotiation by revealing only some of the equilibria she knows about, and can alter the outcome significantly in her favor. But, revealing too few equilibria leads to an outcome that is worse for both agents.

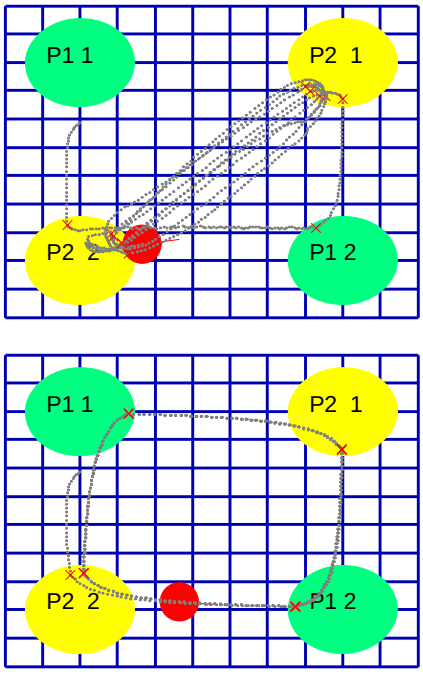

Figure 3. Simulated execution traces for negotiated policies. Top: with 2 points per state in the convex hull, our algorithm settles on a mostly-selfish path. Bottom: with 32 points per state, we find the cooperative path. Steps where either player achieved a goal are marked with $\times$.

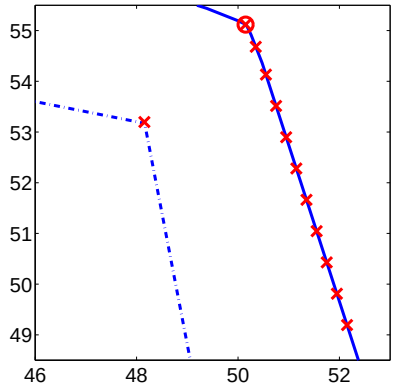

Figure 4. Negotiation between agents with different computational abilities. Solid line: Pareto frontier computed by fast agent; dash-dot line: slow agent's frontier; $\otimes$ mark: outcome of bargaining if fast agent reveals all she knows; $\times$ marks: outcomes if fast agent hides some information.