No-Regret Learning and a Mechanism for Distributed Multiagent Planning

Jan-P. Calliess
Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA, 15213,USA
jcalliess @gmail.com

Geoffrey J. Gordon
Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA, 15213,USA
ggordon @cs.cmu.edu

ABSTRACT

We develop a novel mechanism for coordinated, distributed multiagent planning. We consider problems stated as a collection of single-agent planning problems coupled by common soft constraints on resource consumption. (Resources may be real or fictitious, the latter introduced as a tool for factoring the problem). A key idea is to recast the distributed planning problem as learning in a repeated game between the original agents and a newly introduced group of *adversarial* agents who influence *prices* for the resources. The adversarial agents benefit from *arbitrage*: that is, their incentive is to uncover violations of the resource usage constraints and, by selfishly adjusting prices, encourage the original agents to avoid plans that cause such violations. If all agents employ no-regret learning algorithms in the course of this repeated interaction, we are able to show that our mechanism can achieve design goals such as social optimality (efficiency), budget balance, and Nash-equilibrium convergence to within an error which approaches zero as the agents gain experience. In particular, the agents' average plans converge to a socially optimal solution for the original planning task. We present experiments in a simulated network routing domain demonstrating our method's ability to reliably generate sound plans.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Intelligent Agents, Multiagent Systems*

General Terms

Algorithms, Economics, Design, Theory

Keywords

Multiagent planning, multiagent learning, auctions, mechanism design, game theory, no-regret algorithms

1. INTRODUCTION

In this work, we develop a novel, distributed multiagent planning mechanism. Our mechanism coordinates the different, individual goals of participating agents $P_1, ..., P_k$ to achieve a globally desirable plan. While the agents could in principle compute the optimal global plan in a centralized manner, distributed approaches can improve robustness, fault tolerance, scalability (both in problem complexity and in the number of agents), and flexibility in changing environments [18].

We consider multiagent planning problems stated as $k \in \mathbb{N}$ single-agent convex optimization problems that are coupled by $n \in \mathbb{N}$ linear, soft constraints with positive coefficients. (By a *soft* constraint, we mean that violations are feasible, but are penalized by a convex loss function such as a hinge loss.) This representation includes, for example, network routing problems, in which each agent's feasible region represents the set of paths from a source to a sink, its objective is to find a low-latency path, and a soft constraint represents the additional delay caused by congestion on a link used by multiple agents.

Since all coupling constraints are soft, each agent's feasible region does not depend on the actions of the other agents. So, the agents could plan independently and be guaranteed that their joint actions would be feasible. This interaction is a *convex game* [20]: each player simultaneously selects her plan from a convex set, and if we hold all plans but one fixed, the remaining player's loss is a convex function of her plan. By playing this convex game repeatedly, the players could learn about one another's behavior and adjust their actions accordingly. With appropriate learning algorithms they could even ensure convergence to an equilibrium [3, 12]. Unfortunately, while distributed and robust, this naïve setup can lead to highly suboptimal global behavior: a selfish agent which can gain any benefit from using a congested link will do so, even if the resulting cost to other agents would far outweigh its own gain [17].

Instead, a key idea of our approach is to introduce additional *adversarial* agents $A_1, ..., A_n$, each of which can influence the cost of one of the resources by collecting usage fees. Like the original agents, the new agents are self-interested. But, collectively, they encourage the original agents to avoid excessive resource usage: we show below how to define their revenue functions so that they effectively perform *arbitrage*, allocating extra constraint-violation costs to each original agent in proportion to its responsibility for the violations.

The way we define the adversarial agents' revenues and payments effectively *decouples* the individual planning problems: an original agent perceives the other original agents' actions only through their effects on the choices of the adversarial agents. So, under our mechanism, the original agents never need to communicate with one another directly. In-

Cite as: No-Regret Learning and a Mechanism for Distributed Multiagent Planning, Jan-P. Calliess and Geoffrey J. Gordon, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons (eds.), May, 12-16., 2008, Estoril, Portugal, pp. XXX-XXX.

stead, they communicate with the adversarial agents to find out prices and declare demands for the resources they need to use. If an original agent never uses a particular resource, it never needs to send messages to the corresponding adversarial agent. This decoupling can greatly reduce communication requirements and increase robustness compared to the centralized solution: a central planner must communicate with every agent on every time step, and so constitutes a choke point for communication as well as a single point of failure.

Because we have decoupled the agents from one another, each individual agent no longer needs to worry about the whole planning problem. Instead, it only has to solve a local online convex problem (OCP) [11, 22] independently and selfishly. To solve its OCP, an agent could use any learning algorithm it desired; but, in this paper, we explore what happens if the agents use *no-regret* learning algorithms such as Greedy Projection [22]. No-regret algorithms are a natural choice for agents in a multi-player game, since they provide performance guarantees that other types of algorithms do not. And, as we will show, if all agents commit to no-regret learning, several desirable properties result.

More specifically, if each agent's average per-iteration regret approaches zero, the agents will learn a globally optimal plan, in two senses: first, the *average per-iteration cost*, summed over all of the agents, will converge to the optimal social cost for the original planning problem. And second, the *average overall plan* of the original agents will converge to a socially optimal solution of the original planning problem.

These two results lead us to propose two different mechanism variants: in the **online** setup, motivated by the first result, learning takes place online in the classic sense; all agents choose and execute their plans and make and receive payments in every learning iteration. By contrast, in the **negotiation** setup, motivated by the second result, the agents learn and plan as usual, but only simulate the execution of their chosen joint plan. The simulated results (costs or rewards) from this proposed joint plan provide feedback for the learners. After a sufficient number of learning iterations, each agent averages together all of its proposed plans and executes the average plan. One can interpret either setup, but the negotiation setup in particular, as a very simple auction where the goods are resources. A plan which consumes a resource is effectively a bid for that resource; the resource prices are determined by the agents' learning behavior in response to the bids.

Just as with any mechanism, because our agents are selfish, we need to consider the impact of individual incentives. Focusing on the negotiation setup, we provide (mainly asymptotic) guarantees of Nash-equilibrium convergence of the overall learning outcome as well as classic mechanism design goals such as budget balance, individual rationality, and efficiency.

Due to space constraints, we have omitted some proofs and some discussion of details and variants of our approach. These may be found in the long version of this paper [5].

2. PRELIMINARIES

In an *online convex program* [11, 22], a possibly adversarial sequence $(\Gamma_{(t)})_{t\in\mathbb{N}}$ of convex cost functions is revealed step by step. (Equivalently, one could substitute concave reward functions.) At each step t, the OCP algorithm must choose a play $\mathbf{x}_{(t)}$ from its feasible region F while only knowing the *past* cost functions $\Gamma_{(q)}$ and choices $\mathbf{x}_{(q)}$ $(q \leq t-1)$. After the choice is made, the current cost function $\Gamma_{(t)}$ is revealed, and the algorithm pays $\Gamma_{(t)}(\mathbf{x}_{(t)})$.

To measure the performance of an OCP algorithm, we can compare its accumulated cost up through step T to an estimate of the best cost attainable against the sequence $(\Gamma_{(t)})_{t=1\ldots T}$. Here, we will estimate the best attainable cost as the cost of the best constant play $\mathbf{s}_{(T)} \in F$, chosen with knowledge of $\Gamma_{(1)} \ldots \Gamma_{(T)}$. This choice leads to a measure called *external regret* or just *regret*: $R(T) = \sum_{t=1}^{T} \Gamma_{(t)}(\mathbf{x}_{(t)}) - \sum_{t=1}^{T} \Gamma_{(t)}(\mathbf{s}_{(T)})$. An algorithm is *no-regret* iff it guarantees that $R(T)$ grows slower than $O(T)$, i.e., $R(T) \leq \Delta(T) \in o(T)$. $\Delta(T)$ is a *regret bound.* (The term *no-regret* is motivated by the fact that the limiting average regret is no more than zero, $\limsup_{T\to\infty} R(T)/T \leq 0$.)

We define the *convex conjugate* or *dual* [4] of a function $\Gamma(\mathbf{x})$ to be $\Gamma^*(\mathbf{y}) = \sup_{\mathbf{x}\in \mathrm{dom}\ \Gamma}[\langle \mathbf{x}, \mathbf{y}\rangle - \Gamma(\mathbf{x})]$. The conjugate function Γ^* is always closed and convex, and if Γ is closed and convex, then $\Gamma^{**} = \Gamma$ pointwise.

2.1 Model and Notation

We wish to model interaction among k *player* agents, $P_1 \ldots P_k$. We represent player P_i's individual planning problem as a convex program: choose a vector $\mathbf{p}_i$ from a compact, convex feasible set F_{P_i} to minimize the *intrinsic* cost $\langle \mathbf{c_i}, \mathbf{p_i}\rangle$. (In addition to the intrinsic cost, player P_i will attempt to minimize cost terms which arise from interactions with other agents; we will define these extra terms below and add them to P_i's objective.) We assume that the intrinsic cost vector $\mathbf{c}_i$ and feasible region F_{P_i} are private information for P_i—that is, P_i may choose to inform other players about them, but may also choose to be silent or to lie.

We assume each feasible set F_{P_i} is a subset of some finite-dimensional Hilbert space V_{P_i} with real-valued scalar product $\langle \cdot, \cdot \rangle_{V_{P_i}}$. We write $V = V_{P_1} \times \ldots \times V_{P_k}$ and $F_P = F_{P_1} \times \ldots \times F_{P_k}$ for the *overall* planning space and feasible region, and $\mathbf{p} = (\mathbf{p}_1; \ldots; \mathbf{p}_k)$ and $\mathbf{c} = (\mathbf{c}_1; \ldots; \mathbf{c}_k)$ for the overall plan and combined objective. (We use ; to denote vertical stacking of vectors, consistent with Matlab usage.) And, for any $\mathbf{c}, \mathbf{p} \in V$, we write $\langle \mathbf{c}, \mathbf{p}\rangle_V = \sum_i \langle \mathbf{c_i}, \mathbf{p_i}\rangle_{V_{P_i}}$ for our scalar product on V. (We omit subscripts on $\langle \cdot, \cdot \rangle$ when doing so will not cause confusion.) For convenience, we assume each feasible set only contains vectors with nonnegative components; we can achieve this property by changing coordinates if necessary.

We model the coupling among players by a set of n soft linear constraints, which we interpret as resource consumption constraints. That is, we assume that there are vectors $\mathbf{l}_{ji} \geq 0$ such that the consumption of resource j by plan $\mathbf{p}_i \in V_{P_i}$ is $\langle \mathbf{l_{ji}}, \mathbf{p_i}\rangle$. (So, the total consumption of resource j is $\langle \mathbf{l_j}, \mathbf{p}\rangle$, where $\mathbf{l}_j = (\mathbf{l}_{j1}; \ldots; \mathbf{l}_{jk})$.) And, we assume that there are monotone increasing, continuous, convex penalty functions $\beta_j(\nu)$ and scalars $y_j \geq 0$ so that the overall cost due to consumption of resource j in plan $\mathbf{p}$ is $\beta_j(\langle \mathbf{l_j}, \mathbf{p}\rangle - y_j)$. In keeping with the interpretation of β_j as enforcing a soft constraint, we assume $\beta_j(\nu) = 0$ for all $\nu \leq 0$. (For example, $\beta_j(\nu)$ could be the hinge loss function $\max\{0, \nu\}$.) We define

$$\nu_j(\mathbf{p}) = \langle \mathbf{l_j}, \mathbf{p}\rangle - y_j \qquad (1)$$

to be the magnitude of violation of the jth soft resource constraint by plan $\mathbf{p}$.[1] Because the resource constraints are soft, no player can make another player's chosen action infeasible; conflicts result only in high cost rather than infeasibility.

The function β_j describes the *overall* cost to all players of usage of resource j. We will assume that, in the absence of any external coordination mechanism, the cost to player i due to resource j is given by some function $\beta_{ji}(\mathbf{p})$ with $\sum_i \beta_{ji}(\mathbf{p}) = \beta_j(\nu_j(\mathbf{p}))$. We will call β_{ji} the *natural cost* or *cost in nature* of P_i's resource usage. A typical choice for β_{ji} is proportional to player i's consumption of resource j:

$$\beta_{ji}(\mathbf{p}) = \begin{cases} 0 & \text{if } \langle \mathbf{l_j}, \mathbf{p} \rangle = 0 \\ \beta_j(\nu_j(\mathbf{p}))\langle \mathbf{l_{ji}}, \mathbf{p_i} \rangle / \langle \mathbf{l_j}, \mathbf{p} \rangle & \text{otherwise} \end{cases} \quad (2)$$

So, including both her intrinsic costs and the natural costs of resource usage, player i's objective is

$$\omega_{P_i}(\mathbf{p}) = \langle \mathbf{c_i}, \mathbf{p_i} \rangle + \sum_j \beta_{ji}(\mathbf{p}) \ .$$

We will write $\omega(\mathbf{p}) = \sum_i \omega_{P_i}(\mathbf{p})$ for the social cost; as stated above, our goal is to coordinate the player agents to minimize $\omega(\mathbf{p})$. (With this notation, several facts mentioned above should now be obvious: for example, since the individual objectives ω_{P_i} depend on the entire joint plan $\mathbf{p}$, the players cannot simply plan in isolation. Nor do we want the players to compute and follow an equilibrium: using the above choice for β_{ji} (which results in a setting similar to so-called *nonatomic congestion games*), there are simple sequences of examples showing that the penalty for following an equilibrium (called the *price of anarchy*) can be arbitrarily large [17].)

3. PROBLEM TRANSFORMATION

In this section, by dualizing our soft resource constraints, we decouple the problem of finding a socially optimal plan. The result is a *saddle-point* or *minimax* problem whose variables are the original plan vector $\mathbf{p}$ along with new dual variables $\mathbf{a}$, defined below. By associating the new variables $\mathbf{a}$ with additional agents, called *adversarial agents*, we arrive at a convex game with comparatively-sparse interactions. Based on this game, we introduce our proposed learning-based mechanism.

3.1 Introduction of adversarial agents

Write $F_{A_j} = \text{dom } \beta_j^*$. Since we have assumed that β_j is continuous and convex, we know that $\beta_j^{**} = \beta_j$ pointwise, that is, $\beta_j(\nu) = \sup_{a^j \in F_{A_j}} [a^j \nu - \beta_j^*(a^j)]$ for all $\nu \in \mathbb{R}$. Since β_j^* will become part of the objective function for the adversarial agents, and since many online learning algorithms require compact domains, we will assume that $F_{A_j} = [0, u^j]$ for some scalar u^j. (For example, this assumption is satisfied if the slope of β_j is upper bounded by u_j, and achieves its upper bound. The lower bound of zero follows from our previous assumptions that β_j is monotone and $\beta_j(\nu) = 0$ for $\nu \leq 0$.) We will also assume that β_j^* is continuous on its domain.

We define $\Omega_{A_j} : V \times F_{A_j} \to \mathbb{R}$ as

$$\Omega_{A_j}(\mathbf{p}, a^j) = a^j \nu_j(\mathbf{p}) - \beta_j^*(a^j) \ . \quad (3)$$

And, writing $\mathbf{a} = (a^1; \ldots; a^n) \in F_A = F_{A_1} \times \ldots \times F_{A_n}$, we define

$$\Omega(\mathbf{p}, \mathbf{a}) = \langle \mathbf{c}, \mathbf{p} \rangle + \sum_{j=1}^{n} \Omega_{A_j}(\mathbf{p}, a^j) \quad (4)$$

(note the inclusion of the intrinsic cost $\langle \mathbf{c}, \mathbf{p} \rangle$). Because of the duality identity mentioned above, along with our assumption about dom β_j^*, we know that for all plans $\mathbf{p}$, $\sup_{a^j \in F_{A_j}} \Omega_{A_j}(\mathbf{p}, a_j) = \beta_j(\nu_j(\mathbf{p}))$, and so

$$\omega(\mathbf{p}) = \max_{\mathbf{a} \in F_A} \Omega(\mathbf{p}, \mathbf{a}) \ . \quad (5)$$

Note that we have replaced sup by max in Eq. 5: since $\Omega(\mathbf{p}, \cdot)$ is a closed concave function, it achieves its supremum on a compact domain such as F_A.

Now, as promised, we can introduce the *adversarial agents*: the adversarial agent A_j controls the parameter $a^j \in F_{A_j}$, and tries to maximize its revenue $\Omega_{A_j}(\mathbf{p}, a^j) - \beta_j(\nu_j(\mathbf{p}))$. Note that $\beta_j(\nu_j(\mathbf{p}))$ does not depend on a^j, and so does not affect the choice of a^j once $\mathbf{p}$ is fixed.

To give A_j this revenue, we will have player P_i pay adversary A_j the amount $a^j \langle \mathbf{l_{ji}}, \mathbf{p_i} \rangle - \beta_{ji}(\mathbf{p}) - d_{ji} r_j(a^j)$. Here the *remainder function* r_j is defined as $r_j(a^j) = a^j y_j + \beta_j^*(a^j)$; the nonnegative weights d_{ji} are responsible for dividing up the remainder among all player agents, so we require $\sum_i d_{ji} = 1$ for each j. Given these definitions, it is easy to check that the sum of all payments to A_j is indeed $\Omega_{A_j}(\mathbf{p}, a^j) - \beta_j(\nu_j(\mathbf{p}))$ as claimed.

We can interpret the above payments as follows: A_j sets the per-unit price a^j for consumption of resource j. P_i pays A_j according to consumption, $a^j \langle \mathbf{l_{ji}}, \mathbf{p_i} \rangle$, and is reimbursed for her share of the actual resource cost, $\beta_{ji}(\mathbf{p})$. For the privilege of setting the per-unit price, A_j pays a fee $r_j(a^j)$; this fee is distributed back to the player agents according to weights d_{ji}. (We show in the long version that the fee is always nonnegative.) Since the entire revenue for the agents A_j arises from payments by the player agents, we can think of A_j as opponents for the players—this is qualitatively true even though our game has many players and even though the player agent payoff functions contain terms that do not involve $\mathbf{a}$.

Including payments to adversaries, P_i's cost becomes

$$\Omega_{P_i}(\mathbf{p}_i, \mathbf{a}) = \langle \mathbf{c_i}, \mathbf{p_i} \rangle + \sum_j (a^j \langle \mathbf{l_{ji}}, \mathbf{p_i} \rangle - d_{ji} r_j(a^j)) \quad (6)$$

By the above construction, we have achieved several important properties:

- First, as promised, P_i's cost does not depend on any components of $\mathbf{p}$ other than $\mathbf{p}_i$, and A_j's revenue does not depend on any components of $\mathbf{a}$ other than a^j. So, given an adversarial play $\mathbf{a}$, each player could plan by independently optimizing $\Omega_{P_i}(\mathbf{p}_i, \mathbf{a})$. Similarly, given a plan $\mathbf{p}$, each adversary could separately optimize $\Omega_{A_j}(\mathbf{p}, a_j)$. ($A_j$ can ignore the term $\beta_j(\nu_j(\mathbf{p}))$ if $\mathbf{p}$ is fixed.) So, the players' optimization problems are *decoupled* given $\mathbf{a}$, and the adversaries' optimization problems are *decoupled* given $\mathbf{p}$.

- Second, if an adversarial agent plays optimally, her revenue will be exactly zero, since $\max_{a^j \in [0, u^j]} \Omega_{A_j}(\mathbf{p}, a^j) = \beta_j(\nu_j(\mathbf{p}))$. (Suboptimal play will lead to a negative revenue, i.e., a loss or cost.)

[1] To enforce a hard constraint, we could choose a sufficiently small margin $\epsilon > 0$, replace y_j by $y_j - \epsilon$, and set $\beta_j(\nu) = \max\{0, \nu/\epsilon\}$.

- Third, the total cost to all player agents is
$$\sum_i \Omega_{P_i}(\mathbf{p}_i, \mathbf{a}) = \Omega(\mathbf{p}, \mathbf{a}) \ . \tag{7}$$
On the other hand, the total revenue to all adversaries is $\Omega_A(\mathbf{p}, \mathbf{a}) = \Omega(\mathbf{p}, \mathbf{a}) - \langle \mathbf{c}, \mathbf{p} \rangle - \sum_j \beta_j(\nu_j(\mathbf{p}))$. If the adversaries each play optimally, then $\Omega_A(\mathbf{p}, \mathbf{a})$ will be zero, so we will have
$$\Omega(\mathbf{p}, \mathbf{a}) = \langle \mathbf{c}, \mathbf{p} \rangle + \sum_j \beta_j(\nu_j(\mathbf{p})) = \omega(\mathbf{p}) \ . \tag{8}$$
Combining Eqs. 7 and 8, we find that if the adversaries play optimally, the total cost to the player agents is $\omega(\mathbf{p})$, just as it was in our original planning problem.

- Finally, since $\Omega(\mathbf{p}, \mathbf{a})$ is a continuous saddle-function on a compact domain [16], it must have a saddle-point $(\tilde{\mathbf{p}}, \tilde{\mathbf{a}})$. (By definition, a saddle-point is a point $(\tilde{\mathbf{p}}, \tilde{\mathbf{a}})$ such that $\Omega(\mathbf{p}, \tilde{\mathbf{a}}) \geq \Omega(\tilde{\mathbf{p}}, \tilde{\mathbf{a}}) \geq \Omega(\tilde{\mathbf{p}}, \mathbf{a})$ for all $\mathbf{p} \in F_P$ and $\mathbf{a} \in F_A$.) By the decoupling arguments above, we must have that $\tilde{\mathbf{p}}_i \in \arg\min_{\mathbf{p}_i \in F_{P_i}} \Omega_{P_i}(\mathbf{p}_i, \tilde{\mathbf{a}})$ for each i, and that $\tilde{\mathbf{a}}_j \in \arg\max_{\mathbf{a}_j \in F_{A_j}} \Omega_{A_j}(\tilde{\mathbf{p}}, \mathbf{a}_j)$ for each j. (The latter is true since Ω and Ω_A differ only by terms that do not depend on $\mathbf{a}$.)

3.2 Planning as learning in a repeated game

If we consider our planning problem as a game among all of the agents P_i and A_j, we have just shown that there exists a Nash equilibrium in pure strategies, and that in any Nash equilibrium, the player plan $\tilde{\mathbf{p}}$ must minimize $\omega(\tilde{\mathbf{p}})$ and therefore be socially optimal. To allow the agents to find such an equilibrium, we now cast the planning problem as learning in a repeated game. We will show that, if each agent employs a no-regret learning algorithm, the agents as a whole will converge to a socially optimal plan, both in the sense that the average joint plan converges and in the sense that the average social cost converges. (This result, while similar to well-known results about convergence of no-regret algorithms to minimax equilibrium [8], does not follow from these results, in part because our game is not constant-sum.) Note that, from the individual agent's perspective, playing in the repeated game is an OCP, and so using a no-regret learner would be a reasonable choice; we explore the effect of this choice in more detail below in Sec. 4.

The repeated game is played between the k players and the n adversarial agents. Based on their local histories of past observations, in each round t, each player P_i chooses a current pure strategy $\mathbf{p}_{i(t)} \in F_{P_i}$, and simultaneously, each adversary A_j chooses a current resource price $a^j_{(t)} \in F_A$. We write $\mathbf{p}_{(t)} = (\mathbf{p}_{1(t)}; ...; \mathbf{p}_{k(t)})$ and $\mathbf{a}_{(t)} = (a^1_{(t)}; ...; a^n_{(t)})$ for the joint actions of all players and adversaries, respectively.

After choosing $\mathbf{p}_{(t)}$ and $\mathbf{a}_{(t)}$, the players send their current resource consumptions $\langle \mathbf{p_{i(t)}}, \mathbf{l_{ji}} \rangle$ to the adversaries, and the adversaries send their current prices to the players. In the online model, P_i observes $\beta_{ji}(\mathbf{p}_{(t)})$ and sends it to A_j as well; in the negotiation model, we assume that β_{ji} is of the form given in Eq. 2, so that A_{ji} can compute $\beta_{ji}(\mathbf{p}_{(t)})$. The above information allows each P_i to compute its current cost function $\Omega_{P_i(t)}(\cdot) = \Omega_{P_i}(\cdot, \mathbf{a}_{(t)})$ and its cost $\Omega_{P_i(t)}(\mathbf{p}_{i(t)})$. It also allows each A_j to compute $\Omega_{A_j(t)}(\cdot) = \Omega_{A_j}(\mathbf{p}_{(t)}, \cdot)$ as well as $\beta_{j(t)} = \beta_j(\nu_j(\mathbf{p}_{(t)}))$, and thus, its total revenue $\Omega_{A_j(t)}(a^j_{(t)}) - \beta_{j(t)}$. (In fact, A_j may avoid computing or storing $\beta_{j(t)}$ if desired, since it does not influence that term directly.) In Sec. 3.3 below, we discuss how to implement the necessary communication efficiently.

Each player P_i then adds observation $\mathbf{p}_{(t)}, \Omega_{P_i(t)}(\cdot)$ to her local history, and each adversary A_j adds $\mathbf{a}_{(t)}, \Omega_{A_j(t)}(\cdot)$ to her local history. Finally, the system enters iteration $t+1$, and the process repeats.

3.2.1 Game between two synthesized agents

For analysis, it will help to construe our setup as a fictitious game between a *synthesized player agent*, P, and a *synthesized adversarial agent*, A. When each component agent P_i plays $\mathbf{p}_{i(t)}$ and each component agent A_j plays $a^j_{(t)}$, then we imagine P to play $\mathbf{p}_{(t)}$ and A to play $\mathbf{a}_{(t)}$. Accordingly, we understand P to incur cost $\Omega(\mathbf{p}_{(t)}, \mathbf{a}_{(t)})$, and A to have revenue $\Omega_A(\mathbf{p}_{(t)}, \mathbf{a}_{(t)}) - \sum_j \beta_j(\nu_j(\mathbf{p}_{(t)}))$ in round t. These synthesized agents are merely theoretical notions serving to simplify our reasoning; in practice there would never be a single agent controlling all players.

Using these synthesized agents, we will prove two results: first, immediately below, we show that if the individual agents use no-regret algorithms, then the synthesized agents also achieve no regret. And second, in Sec. 3.2.2, we show that if the synthesized agents achieve no regret, then they will converge to an equilibrium of the game, in the two senses mentioned above.

Lemma 3.1. *If each individual agent P_i achieves regret bound $\Delta_{P_i}(T)$, then the synthesized player agent P achieves regret bound $\Delta_P(T) := \sum_i \Delta_{P_i}(T)$. So, if $\Delta_{P_i}(T) \in o(T)$ for all i, then $\Delta_P(T) \in o(T)$.*

Proof. By definition, the regret $R_P(T)$ for agent P is
$$R_P(T) = \textstyle\sum_{t=1}^T \Omega(\mathbf{p}_{(t)}, \mathbf{a}_{(t)}) - \min_{\mathbf{p}} \sum_{t=1}^T \Omega(\mathbf{p}, \mathbf{a}_{(t)})$$
and the regret for P_i is $R_{P_i}(T) \leq \Delta_{P_i}(T)$:
$$R_{P_i}(T) = \textstyle\sum_{t=1}^T \Omega_{P_i(t)}(\mathbf{p}_{i(t)}) - \min_{\mathbf{p}_i} \sum_{t=1}^T \Omega_{P_i(t)}(\mathbf{p}_i)$$
Owing to the decoupling effect of the adversary we have $\Omega(\mathbf{p}, \mathbf{a}_{(t)}) = \sum_{i=1}^k \Omega_{P_{i(t)}}(\mathbf{p}_i)$. So, we can expand $R_P(T)$ as
$$\begin{aligned}
&\sum_{t=1}^T \Omega(\mathbf{p}_{(t)}, \mathbf{a}_{(t)}) - \min_{\mathbf{p} \in F_P} \sum_{t=1}^T \Omega(\mathbf{p}, \mathbf{a}_{(t)}) \\
&\quad = \sum_{t=1}^T \sum_{i=1}^k \Omega_{P_{i(t)}}(\mathbf{p}_{i(t)}) - \min_{\mathbf{p} \in F_P} \sum_{t=1}^T \sum_{i=1}^k \Omega_{P_{i(t)}}(\mathbf{p}_i) \\
&\quad = \sum_{i=1}^k \sum_{t=1}^T \Omega_{P_{i(t)}}(\mathbf{p}_{i(t)}) - \sum_{i=1}^k \min_{\mathbf{p}_i \in F_{P_i}} \sum_{t=1}^T \Omega_{P_{i(t)}}(\mathbf{p}_i) \\
&\quad = \sum_{i=1}^k (\sum_{t=1}^T \Omega_{P_{i(t)}}(\mathbf{p}_{i(t)}) - \min_{\mathbf{p}_i \in F_{P_i}} \sum_{t=1}^T \Omega_{P_{i(t)}}(\mathbf{p}_i)) \\
&\quad = \sum_{i=1}^k R_{P_i}(T) \ \leq \ \sum_{i=1}^k \Delta_{P_i}(T) \ .
\end{aligned}$$
So, $R_P(T) \in o(T)$ as desired. □

Analogously, for the adversarial agent A we have the following lemma. The proof is very similar, and is therefore omitted.

Lemma 3.2. *If each adversarial agent A_j achieves regret bound $\Delta_{A_j}(T)$, then the synthesized agent A achieves regret* *bound* $\Delta_A(T) := \sum_j \Delta_{A_j}(T)$. *So, if* $\Delta_{A_j}(T) \in o(T)$ *for all* j*, then* $\Delta_A(T) \in o(T)$.

3.2.2 Social optimality

In this section, we investigate the behavior of the averaged strategies $\bar{\mathbf{p}}_{[T]} := \frac{1}{T}\sum_{t=1}^{T} \mathbf{p}_{(t)}$ and $\bar{\mathbf{a}}_{[T]} := \frac{1}{T}\sum_{t=1}^{T} \mathbf{a}_{(t)}$, as well as the averaged costs $\frac{1}{T}\sum_{t=1}^{T} \Omega(\mathbf{p}_{(t)}, \mathbf{a}_{(t)})$. (Recall that the negotiation version of our mechanism outputs the averaged strategies, while the online version of our mechanism incurs the averaged costs.)

Starting with the averaged strategies, we show that if all players achieve no regret, we can guarantee convergence of $(\bar{\mathbf{p}}_{[T]}, \bar{\mathbf{a}}_{[T]})$ to a set $K_P \times K_A$ of saddle-points of Ω. (While the sequences $\bar{\mathbf{p}}_{[T]}$ and $\bar{\mathbf{a}}_{[T]}$ may not converge, the distance of $\bar{\mathbf{p}}_{[T]}$ from K_P and the distance of $\bar{\mathbf{a}}_{[T]}$ from K_A will approach zero; and, every cluster point of the sequence $(\bar{\mathbf{p}}_{[T]}, \bar{\mathbf{a}}_{[T]})$ will be in $K_P \times K_A$. Due to the convexity and compactness of the feasible sets, each average strategy and each cluster point will be feasible.)

THEOREM 3.3. *Let* $\bar{\mathbf{p}}_{[T]} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{p}_{(t)}$, $\bar{\mathbf{a}}_{[T]} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{a}_{(t)}$ *be the averaged, pure strategies of synthesized player* P *and adversary* A*, respectively. If* P*,* A *each suffer sublinear external regret, then as* $T \to \infty$*,* $(\bar{\mathbf{p}}_{[T]}, \bar{\mathbf{a}}_{[T]})$ *converges to a (bounded) subset* $K_P \times K_A$ *of saddle-points of the player cost function* Ω.

Since Ω is continuous on its domain, Thm. 3.3 lets us conclude that the outcome of negotiation is a plan which approximately minimizes total player cost in nature: if we choose T sufficiently large, then $(\bar{\mathbf{p}}_{[T]}, \bar{\mathbf{a}}_{[T]})$ must be close to a saddle point, and so the costs must be close to the costs of a saddle point. (To determine how large we need to choose T, we can look at the regret bounds of the learning algorithms of the individual agents.)

If we choose to run our mechanism in online mode, we also need bounds on the average incurred cost. Since $\omega(\mathbf{p}) = \max_{\mathbf{a}} \Omega(\mathbf{p}, \mathbf{a})$, the following theorem tells us that the average social cost approaches the optimal social cost in the long run.

THEOREM 3.4. *If* P *and* A *suffer sublinear external regret, then as* $T \to \infty$*,*

$$\frac{1}{T}\sum_{t=1}^{T} \Omega(\mathbf{p}_{(t)}, \mathbf{a}_{(t)}) \to \min_{\mathbf{p}} \max_{\mathbf{a}} \Omega(\mathbf{p}, \mathbf{a}) \ .$$

The proofs of Theorems 3.3 and 3.4 appear in the long version of this paper [5].

3.3 Communication costs

So far we have assumed that all player agents broadcast their resource usages (and possibly their natural costs) to all adversarial agents, and all adversarial agents broadcast their prices to all player agents. With this assumption, on every time step, each player sends one broadcast of size $O(n)$ (her resource usages) and receives n messages of size $O(1)$ (the resource prices), while each adversary sends one broadcast of size $O(1)$ and receives k messages of size $O(n)$, for a total of $n + k$ broadcasts per step, and a total incoming bandwidth of no more than $O(nk)$ at each agent.[2] Even under this simple assumption, the cost is somewhat better than a centralized planner, which would have to receive k much-larger messages describing each player's detailed optimization problem, and send k much-larger messages describing each player's optimal plan.

However, by exploiting *locality*, we can reduce bandwidth even further: in many problems we can guarantee *a priori* that player P_i will never use resource j, and in this case, we never need to transmit A_j's price a^j to P_i. Similarly, if P_i decides not to use resource j on a given trial, we never need to transmit $\langle \mathbf{l_{ji}}, \mathbf{p_{i(t)}} \rangle$ to A_j. (To take full advantage of locality, we must also set the weights d_{ji} so that players do not receive payments from adversaries they would otherwise not need to talk to.) So, by using targeted multicasts instead of broadcasts, we can confine each player's messages to a small area of the network; in this case, no single node or link will see even $O(k+n)$ traffic. We can sometimes reduce bandwidth even further by combining messages as they flow through the network: for example, two resource consumption messages destined for A_j may be combined by adding their reported consumption values.

Finally, any implementation needs to make sure that the agents cannot gain by circumventing the mechanism: e.g., no player should find out another's plan before committing to her own. In the long version we discuss how to prevent circumvention by encrypting messages where appropriate.

4. DESIGN GOALS

In designing our mechanism, we hope to ensure that individual, incentive-driven behavior leads to desirable system-wide properties. Here, we establish some useful guarantees for the *negotiation version* of our mechanism. The guarantees are **convergence to Nash equilibrium**, **budget balance**, **individual rationality**, and **efficiency**.

These guarantees follow from the fact that the learned negotiation outcome approaches a set of saddle-points of $\Omega(\mathbf{p}, \mathbf{a})$ in the limit (Thm. 3.3). By continuity of Ω, we can therefore conclude that, if we allow sufficient time for negotiation, the negotiation outcome is approximately a saddle-point. (We will not address distributed detection of convergence, but merely assume that we use our global regret bounds to calculate a sufficiently large T ahead of time; obviously efficiency could be improved by allowing early stopping.)

Convergence to Nash equilibrium. When working with selfish, strategic agents, we want to know whether a selfish agent has an incentive to unilaterally deviate from its part of the negotiation outcome. The following theorems show that the answer is, at least approximately, *no*: in the limit of large T, the negotiation outcomes $\bar{\mathbf{p}}_{[T]}, \bar{\mathbf{a}}_{[T]}$ converge to a subset of Nash equilibria. So, by continuity, $(\bar{\mathbf{p}}_{[T]}, \bar{\mathbf{a}}_{[T]})$ is an approximate Nash equilibrium for sufficiently large T—that is, each agent has a vanishing incentive to deviate unilaterally.

THEOREM 4.1. *Let* F_{P_i} *denote the feasible set of player agent* P_i $(i \in \{1, ..., k\})$ *and* F_{A_j} *denote the feasible set of adversarial agent* A_j $(j \in \{1, ..., n\})$*. We have:*

$$\forall i \forall \mathbf{p}'_i \in F_{P_i} \forall \tilde{\mathbf{a}} \in K_A, \tilde{\mathbf{p}} \in K_P : \Omega_{P_i}(\mathbf{p}'_i, \tilde{\mathbf{a}}) \geq \Omega_{P_i}(\tilde{\mathbf{p}}_{\mathbf{i}}, \tilde{\mathbf{a}}).$$

PROOF. Let $\tilde{\mathbf{p}} \in K_P, \tilde{\mathbf{a}} \in K_A$. We know $(\tilde{\mathbf{p}}, \tilde{\mathbf{a}})$ is a saddle-point of Ω with respect to minimization over F_P and maximization over F_A. Hence, $\forall \mathbf{p}' \in F_P : \Omega(\tilde{\mathbf{p}}, \tilde{\mathbf{a}}) \leq \Omega(\mathbf{p}', \tilde{\mathbf{a}})$. Since $\Omega(\mathbf{p}, \mathbf{a}) = \sum_m \Omega_{P_m}(\mathbf{p_m}, \mathbf{a}), \forall \mathbf{p}, \mathbf{a}$, we have in particular: $\forall \mathbf{p}'_i \in$

[2] Technically, the agents could multicast rather than broadcast, so that, e.g., one player would never see another player's messages, but in practice one would not expect this optimization to save much.

$F_{P_i} : \Omega_{P_i}(\mathbf{p}'_{\mathbf{i}}, \tilde{\mathbf{a}}) + \sum_{m \neq i} \Omega_{P_m}(\tilde{\mathbf{p}}_{\mathbf{m}}, \tilde{\mathbf{a}}) = \Omega(\mathbf{p}'_i, \tilde{\mathbf{p}}_{\neg \mathbf{i}}, \tilde{\mathbf{a}}) \geq \Omega(\tilde{\mathbf{p}}, \tilde{\mathbf{a}}) = \Omega_{P_i}(\tilde{\mathbf{p}}_{\mathbf{i}}, \tilde{\mathbf{a}}) + \sum_{m \neq i} \Omega_{P_m}(\tilde{\mathbf{p}}_{\mathbf{m}}, \tilde{\mathbf{a}})$. Thus, $\Omega_{P_i}(\mathbf{p}'_i, \tilde{\mathbf{a}}) \geq \Omega_{P_i}(\tilde{\mathbf{p}}_{\mathbf{i}}, \tilde{\mathbf{a}})$, $\forall \mathbf{p}'_i \in F_{P_i}$. □

THEOREM 4.2. *Let $(\tilde{\mathbf{p}}, \tilde{\mathbf{a}})$ be a saddle-point of $\Omega(\mathbf{p}, \mathbf{a})$ with respect to minimizing over $\mathbf{p}$ and maximizing over $\mathbf{a}$. We have: $\Omega_{A_j}(\tilde{a}^j, \tilde{\mathbf{p}}) = \max_{a^j \in F_{A_j}} \Omega_{A_j}(a^j, \tilde{\mathbf{p}}), \forall j \in \{1, ..., n\}$.*

PROOF. Since $\sum_{j=1}^{n} \Omega_{A_j}(\tilde{a}^j, \tilde{\mathbf{p}}) = \Omega_A(\tilde{\mathbf{a}}, \tilde{\mathbf{p}}) = \max_{\mathbf{a}} \Omega_A(\mathbf{a}, \tilde{\mathbf{p}})$ $= \max_{\mathbf{a} \in F_A} \sum_{j=1}^{n} \Omega_{A_j}(\tilde{a}^j, \tilde{\mathbf{p}}) = \sum_{j=1}^{n} \max_{a^j \in F_{A_j}} \Omega_{A_j}(a^j, \tilde{\mathbf{p}})$, we have $\sum_{j=1}^{n} (\max_{a^j \in F_{A_j}} [\Omega_{A_j}(a^j, \tilde{\mathbf{p}})] - \Omega_{A_j}(\tilde{a}^j, \tilde{\mathbf{p}})) = 0$. On the other hand, $\forall j : \max_{a^j \in F_{A_j}} [\Omega_{A_j}(a^j, \tilde{\mathbf{p}})] - \Omega_{A_j}(\tilde{a}^j, \tilde{\mathbf{p}}) \geq 0$. Hence, $\forall j : \max_{a^j \in F_{A_j}} [\Omega_{A_j}(a^j, \tilde{\mathbf{p}})] - \Omega_{A_j}(\tilde{a}^j, \tilde{\mathbf{p}}) = 0$. □

Theorem 4.2 allows us to conclude that the individual part of an adversarial agent's negotiation outcome is a best-response action:

COROLLARY 4.3. *Let F_{A_j} denote the feasible set of adversarial agent A_j ($j \in \{1, ..., n\}$) and F_{P_i} denote the feasible set of player agent P_i ($i \in \{1, ..., k\}$). We have:*

$$\forall j \forall a^j \in F_{A_j} \forall \tilde{\mathbf{p}} \in K_P, \tilde{\mathbf{a}} \in K_A : \Omega_{A_j}(a^j, \tilde{\mathbf{p}}) \leq \Omega_{A_j}(\tilde{a}^j, \tilde{\mathbf{p}}).$$

Budget balance. Since our overall goal is a socially optimal plan, we would hope that our mechanism neither siphons off money from the agents by running a surplus, nor requires continuous investment to fund a deficit. This is the question of budget balance. Since the agents make payments only to one another (and not directly to the mechanism), in one sense our mechanism is trivially budget balanced. However, a more interesting question is whether the mechanism is budget balanced if we consider the adversarial agents to be part of the mechanism—this additional property guarantees that the adversarial agents do not, in the long run, siphon off money or require external funding. Since we showed (in Sec. 3.1) that the adversarial agents each have zero revenue at any saddle point, and since the outcome of negotiation is an approximate saddle point, our mechanism is (approximately) budget balanced in this sense as well.

Budget balance can be evaluated *ex ante*, *ex interim*, or *ex post*, depending on whether it holds (in expectation) before the agents know their private information, after they know their private information but before they know the outcome of the mechanism, or after they know the outcome of the mechanism. Ex-post budget balance is the strongest property; our argument in fact shows approximate ex-post budget balance.

Individual rationality. Strategic agents will avoid participating in a mechanism if doing so improves their payoffs. A mechanism is individually rational if each agent is no worse off when joining the mechanism than when avoiding it. Just as with budget balance, we can speak of ex-ante, ex-interim, or ex-post individual rationality.

To make the question of individual rationality well-defined, we need to specify what happens if an agent avoids the mechanism. If an adversarial agent refuses to participate, we will assume that her corresponding resource goes unmanaged: no price is announced for it, and the player agents pay their natural costs for it. The adversarial agent therefore gets no revenue, either positive or negative. If a player agent refuses to participate, we will assume that she is constrained use no resources, that is, $\langle \mathbf{l_{ji}}, \mathbf{p_i} \rangle = 0$ for all j. (So, we assume that there is a plan satisfying these constraints.)

Since we showed that $\sup_{a^j} [\Omega_{A_j}(\mathbf{p}, a^j) - \beta_j(\nu_j(\mathbf{p}))] = 0$ (in Sec. 3.1), A_j has (approximately) no incentive to avoid the mechanism when we play an (approximate) saddle point. So, the mechanism is approximately ex-post IR for adversaries.

If a player agent does not participate in the mechanism, she has no chance of acquiring any resources. Since she would not have to pay for joining and using no resources (the remainder $r_j(a^j)$ is nonnegative), it is irrational not to join. So, the mechanism is ex-post IR for players.

Efficiency. A mechanism is called *efficient* if its outcome minimizes global social cost. Thm. 3.3 showed that the mechanism finds an approximate saddle-point of Ω. We showed in Sec. 3.1 that, in any saddle-point, the player cost is $\min_{\mathbf{p}} \omega(\mathbf{p})$ (the socially-optimal cost), and the adversary cost is 0. So, in an approximate saddle-point, the social cost is approximately optimal; the mechanism is therefore approximately efficient.

5. RELATED WORK

The idea of using no-regret algorithms to solve OCPs in order to accomplish a planning task is not new (e.g., [1]). It has, for instance, been proposed for online routing in the Wardrop setting of multi-commodity flows [2], where the authors established convergence to Nash equilibrium for infinitesimal agents. In contrast with this line of work, we seek *globally* good outcomes, rather than just equilibria.

Another body of related work is concerned with selfish routing in *nonatomic* settings (with infinitesimal agents—e.g., [2, 17]). Many of these works provide strong performance guarantees and price of anarchy results considering selfish agents. We consider a similar but not identical setup, with a finite number of agents and divisible resources.

As mentioned before, our planning approach can be given a simple market interpretation: interaction among player agents happens indirectly through resource prices learned by the adversaries. Many researchers have demonstrated experimental success for market-based planners (e.g., [18, 9, 19, 21, 10, 15]). While these works experimentally validate the usefulness of their approaches and implement distributivity, only a few provide guarantees of optimality or approximate optimality (e.g., [14, 21]).

Guestrin and Gordon proposed a decentralized planning method using a distributed optimization procedure based on Benders decomposition [13]. They showed that their method would produce approximately optimal solutions and offered bounds to quantify the quality of this approximation. However, as with most authors, they assumed agents to be *obedient*, i.e., to follow the protocol in every aspect. By contrast, we address *strategic* agents, i.e., selfish, incentive-driven entities prone to deviating from prescribed behavior if it serves their own benefit. But, since the trick of dualizing constraints to decouple an optimization problem is analogous to Benders decomposition, we can view our mechanism as a generalization of Guestrin and Gordon's method to decentralized computation on selfish agents.

Designing systems that provably achieve a desired global behavior with strategic agents is exactly the field of study of classic mechanism design. Many mechanisms, though, are heavy-weight and centralized, and are concerned neither with distributed implementation nor with computational feasibility. Attempting to fill this gap, a new strand of work under the label *distributed algorithmic mechanism* *design* has evolved [7, 6, 21].

Our approach combines many advantages of the above branches of work for multiagent planning. It is distributed, and provides asymptotic guarantees regarding mechanism design goals such as budget balance and quality of the learned solution. If we consider the adversarial agents to be independent, selfish entities that are not part of the mechanism, the proposed mechanism is relatively light-weight; it merely offers infrastructure for the participating agents to coordinate their planning efforts through learning in a repeated game. And, as the following section shows, its theoretical guarantees translate into reliable practical performance, at least in our small-scale network routing experiments.

6. EXPERIMENTS

We conducted experiments on a small multi-agent mincost routing domain. We model our network by a finite, directed graph with edges E (physical links) and vertices V (routers). Each edge $e \in E$ has a finite capacity $\gamma(e)$, as well as a fixed intrinsic cost c^e for each unit of traffic routed through e. We assume that bandwidth is infinitely divisible.

Players are indexed by a source vertex s and a destination vertex r. Player P_{sr} wants to send an amount of flow d_{sr} from s to r. P_{sr}'s individual plan is a vector $\mathbf{f}_{sr} = (f^e_{sr})_{e \in E}$, where $f^e_{sr} \in [0, U]$ is the amount of traffic that P_{sr} routes through edge e. (U is an upper bound, chosen ahead of time to be larger than the largest expected flow.) Feasible plans are those that satisfy *flow conservation*, i.e., the incoming traffic to each vertex must balance the outgoing traffic. We also excluded plans which route flow in circles.

If the total usage of an edge e exceeds its capacity, all agents experience an increased cost for using e. This extra cost could correspond to delays, or to surcharges from a network provider. We set the global penalty for edge e to be a hinge loss function $\beta_e(\nu) = \max\{0, u_e \nu\}$, so the total cost of e increases linearly with the amount of overuse. For convenience we also modeled each player's demand d_{sr} as a soft constraint $\beta_{sr}(\nu) = \max\{0, u_{sr}\nu\}$, although doing so is not necessary to achieve a distributed mechanism.

Applying our problem transformation led to new, total player cost $\Omega(\mathbf{f}, \mathbf{a}) = \sum_{s,r} \sum_{e \in E} c^e f^e_{sr} + \sum_{e \in E} \Omega_{A_e}(a^e_{\text{cap}}, \mathbf{f}) + \sum_{s,r} \Omega_{A_{sr}}(a^{sr}_d, \mathbf{f})$ in the mechanism. Here, for each edge e, we introduced an adversarial agent A_e, who controls the cost for capacity violations at e by setting the price a^e_{cap}. And, as a slight extension to our general description in Section 3, we introduced additional adversarial agents to implement the soft constraints on demand; agent A_{sr} chooses a price a^{sr}_d for failing to meet demand on route sr.

With this setup, we ran more than 2800 simulations, for the most part on random problem instances, but also for manually-designed problems on graphs of sizes varying between 2 and 16 nodes. In each instance there were between 1 and 32 player agents. For no-regret learning, we used the Greedy Projection algorithm [22] with $(\frac{1}{\sqrt{t}})_{t \in \mathbb{N}}$ as the sequence of learning rates.

A simple example of an averaged player plan after a number of iterations is depicted in Figure 1(d). In this experiment, we had a 6-node network and three players $P_{2,3}$, $P_{1,4}$, and $P_{4,6}$ with demands 30, 70 and 110. We set $c^{(2,3)}, c^{(3,2)} = 10$, and $c^e = 1$ for all other edges e. Edges $(5, 6)$ and $(6, 5)$ had capacities of 50, while all other capacities were 100. Our method successfully discovered that $P_{4,6}$ should send as much flow as possible through the cheap edge $(5, 6)$, and the rest along the expensive path through $(3, 2)$. Adversarial agents successfully discouraged the players from violating the capacity constraints, while simultaneously making sure that as much demand as possible was satisfied. Also note that player $P_{2,3}$ served the common good by (on average) routing flow through the pricey edge $(2, 3)$ instead of taking the path through the bottleneck $(6, 5)$; this latter path would have been cheaper for $P_{2,3}$ if we didn't consider the extra costs imposed by the adversarial agents.

The plots in Figs. 1(a)–(c) validate our theoretical results: Figs. 1(a)–(b) demonstrate that the regrets of the combined agents P and A converge to zero, as shown in Lem. 3.1 and Lem. 3.2. Fig. 1(c) demonstrates convergence to a saddle-point of Ω. In the plot, the upper curve shows $\max_{\mathbf{a}} \Omega(\bar{\mathbf{f}}_{[T]}, \mathbf{a})$, while the lower curve shows $\min_{\mathbf{f}} \Omega(\mathbf{f}, \bar{\mathbf{a}}_{[T]})$. The horizontal line is the minimax value of Ω. As guaranteed by Thm. 3.3, the three curves converge to one another. While the fact of convergence in Figs. 1(a)–(c) is not a surprise, it is reassuring to see that the convergence is fast in practice as well as in theory.

7. DISCUSSION

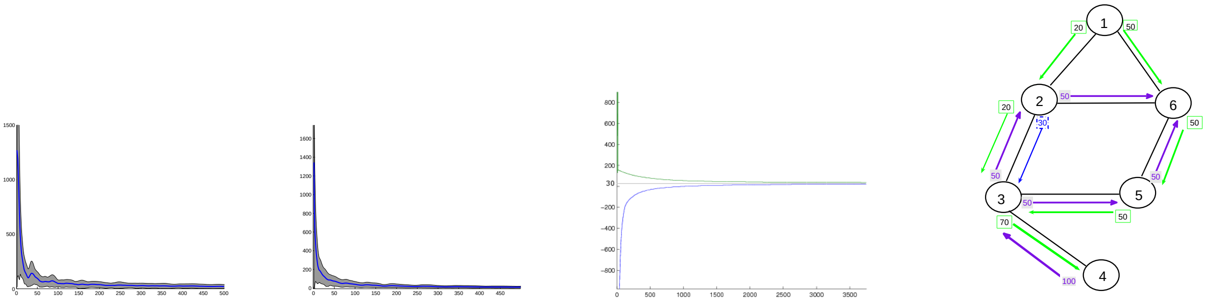
We presented a distributed learning mechanism for use in multiagent planning. The mechanism works by introducing *adversarial* agents who set taxes on common resources. By so doing, it *decouples* the original player agents' planning problems. We then proposed that the original and adversarial agents should learn about one another by playing the decoupled planning game repeatedly, either in reality (the *online* setup) or in simulation (the *negotiation* setup).

We established that, if all agents use no-regret learning algorithms in this repeated game, several desirable properties result. These properties included convergence of $\bar{\mathbf{p}}_{[T]}$, the average composite plan, to a socially optimal solution of the original planning problem, as well as convergence of $\bar{\mathbf{p}}_{[T]}$ and the corresponding adversarial tax-plan $\bar{\mathbf{a}}_{[T]}$ to a Nash equilibrium of the game. We also showed that our mechanism is budget-balanced in the limit of large T.

So far, we do not know in what cases our mechanism is incentive-compatible; in particular, we do not know when it is rational for the individual agents to employ no-regret learning algorithms. Certainly, we can invent cases where it is not rational to choose a no-regret algorithm, but we believe that there are practical situations where no-regret algorithms are a good choice. Investigating this matter, and modifying the mechanism to ensure incentive compatibility in all cases, is left to future work.

Compared to a centralized planner, our method can greatly reduce the bandwidth needed at the choke-point agent. (The choke-point agent is the one who needs the most bandwidth; in a centralized approach it is normally the centralized planner.) In very large systems, agents P_i and A_j only need to send messages to one another if P_i considers using resource j, so we can often use locality constraints to limit the number of messages we need to send.

Our method combines desirable features from various previous approaches: like centralized mechanisms and some other distributed mechanisms we can provide rigorous guarantees such as social optimality and individual rationality. But, like prior work in market-based planning, we expect our approach to be efficient and implementable in a distributed setting. Our experiments tend to confirm this prediction.



(a) (b) (c) (d)

Figure 1: (a), (b): Average positive regret of synthesized agents, averaged over 100 random problems, as a function of iteration number. Grey area indicates standard error. (c): Payoff Ω for the synthesized player (upper curve) and adversary (lower curve) when P and A play their averaged strategy against a best-response opponent in each iteration. Horizontal line shows minimax value. (d): Average plan for three agents in a 6-node instance after 10,000 iterations, rounded to integer flows. The displayed plan is socially optimal.

Acknowledgements

This research was funded in part by a grant from DARPA's Computer Science Study Panel program. All opinions and conclusions are the authors'.

8. REFERENCES

[1] N. Bansal, A. Blum, S. Chawla, and A. Meyerson. Online oblivious routing. In *SPAA '03: Proceedings of the fifteenth annual ACM symposium on Parallel algorithms and architectures*, pages 44–49, 2003.

[2] A. Blum, E. Even-Dar, and K. Ligett. Routing without regret: on convergence to nash equilibria of regret-minimizing algorithms in routing games. In *PODC '06: Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, pages 45–52, 2006.

[3] A. Blum, M. T. Hajiaghayi, K. Ligett, and A. Roth. Regret minimization and the price of total anarchy. Working paper, 2007.

[4] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[5] J.-P. Calliess and G. J. Gordon. No-regret learning and a mechanism for distributed multi-agent planning. Technical Report CMU-ML-08-102, Carnegie Mellon, 2008.

[6] J. Feigenbaum, C. Papadimitriou, R. Sami, and S. Shenker. A BGP-based mechanism for lowest-cost routing. *Distrib. Comput.*, 18(1):61–72, 2005.

[7] J. Feigenbaum, C. H. Papadimitriou, and S. Shenker. Sharing the cost of multicast transmissions. *J. Comput. Syst. Sci.*, 63(1):21–41, 2001.

[8] Y. Freund and R. E. Shapire. Game theory, on-line prediction and boosting. In *COLT*, 1996.

[9] B. Gerkey and M. Mataric. Sold!: Auction methods for multirobot coordination. *IEEE Transactions on Robotics and Automation*, 19(5):758–768, 2002.

[10] E. Gomes and R. Kowalczyk. Reinforcement learning with utility-aware agents for market-based resource allocation. In *AAMAS*, 2007.

[11] G. J. Gordon. Regret bounds for prediction problems. In *COLT: Workshop Comp. Learning Theory*, 1999.

[12] G. J. Gordon, A. Greenwald, C. Marks, and M. Zinkevich. No-regret learning in convex games. Technical Report CS-07-10, Brown University, 2007.

[13] C. Guestrin and G. Gordon. Distributed planning in hierarchical factored MDPs. In *UAI*, 2002.

[14] M. Lagoudakis, V. Markakis, D. Kempe, P. Keskinocak, S. Koenig, A. Kleywegt, C. Tovey, A. Meyerson, and S. Jain. Auction-based multi-robot routing. In *Proceedings of the International Conference on Robotics: Science and Systems (ROBOTICS)*, pages 343–350, 2005.

[15] N. Muguda, P. R. Wurman, and R. M. Young. Experiments with planning and markets in multi-agent systems. *SIGecom Exchanges*, 5:34–47, 2004.

[16] R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

[17] T. Roughgarden. Selfish routing and the price of anarchy (survey). In *OPTIMA*, 2007.

[18] A. Stentz and M. B. Dias. A free market architecture for coordinating multiple robots. Technical Report CMU-RI-TR-99-42, Carnegie Mellon, Robotics Institute, Pittsburgh, PA, USA, 1999.

[19] A. T. Stentz, M. B. Dias, R. M. Zlot, and N. Kalra. Market-based approaches for coordination of multi-robot teams at different granularities of interaction. In *Proc. ANS 10th Int. Conf. on Robotics and Rem. Sys. for Hazardous Env.*, 2004.

[20] G. Stoltz and G. Lugosi. Learning correlated equilibria in games with compact sets of strategies. *Games and Economic Behavior*, 59:187–208, 2007.

[21] M. P. Wellman. A market-orient programming environment and its application to distributed multi-commodity flow problems. *J. of Artifical Intelligence Research*, 1:1–23, 1993.

[22] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Twentieth International Conference on Machine Learning*, 2003.