

A Unification of Extensive-Form Games and Markov Decision Processes

H. Brendan McMahan* and Geoffrey J. Gordon†

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

We describe a generalization of extensive-form games that greatly increases representational power while still allowing efficient computation in the zero-sum setting. A principal feature of our generalization is that it places arbitrary convex optimization problems at decision nodes, in place of the finite action sets typically considered. The possibly-infinite action sets mean we must “forget” the exact action taken (feasible solution to the optimization problem), remembering instead only some statistic sufficient for playing the rest of the game optimally. Our new model provides an exponentially smaller representation for some games; in particular, we show how to compactly represent (and solve) extensive-form games with outcome uncertainty and a generalization of Markov decision processes to multi-stage adversarial planning games.

Introduction

Extensive-form games (EFGs) are commonly used to reason about multiagent interaction, and Markov Decision Processes (MDPs) are commonly used to model single-agent planning in domains where actions have stochastic outcomes. Hundreds of papers have been published on these topics. This paper introduces *Convex Extensive-Form Games* (CEFGs), a powerful generalization of both models that maintains computational tractability. Like an EFG, a CEFG is a game with partial information played on a tree, however, in CEFGs: 1) An arbitrary subset of the players simultaneously select an action at each node; 2) The set of actions available to each player is a convex subset of \mathbb{R}^n , rather than a discrete set; 3) Payoffs are made at internal nodes as well as at leaves, and are given by a bilinear function of the players’ actions; and, 4) Two nodes that are both in the same information set may have different numbers of successor nodes. These differences allow us to embed arbitrary convex optimization problems at nodes of the CEFG, for example allowing us to model MDPs where an opponent controls the cost function.

We first consider the class of bilinear-payoff convex games (CGs), a natural generalization of matrix games.

* mcmahan@google.com. Current affiliation: Google, Inc., CIC building, 4720 Forbes Ave, Pittsburgh, PA 15213.

† ggordon@cs.cmu.edu
Copyright © 2007, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Zero-sum CGs can be solved efficiently via convex optimization, and in fact known algorithms for solving zero-sum EFGs work by reducing the EFG to a CG. This approach to solving EFGs depends on the property of *perfect recall*; the problem is NP-hard without this assumption (Koller & Megiddo 1992). “Perfect recall” CEFGs would not be useful due to the intractable number of possible actions at each node. We develop a generalization of perfect recall for CEFGs, *sufficient recall*, that allows some “forgetting” of past actions. A principal contribution of this work is showing that zero-sum sufficient-recall CEFGs can be transformed to CGs, and hence solved efficiently.

The ability to embed linear programs inside CEFGs unifies this class with MDPs: an MDP is a single-player, single-node CEFG. The problem of solving an MDP where one player selects a policy and another player chooses the cost function was addressed in (McMahan, Gordon, & Blum 2003). This problem can be modeled as a two-player, single-node CEFG. More general versions of this problem, where the players have some limited opportunities to observe their opponent’s past actions, can also be solved as CEFGs. We use a small example of such a multi-stage adversarial path planning problem to illustrate the CEFG model. Our work differs from other work on multi-agent MDPs (Petrik & Zilberstein 2007, for example), in that we focus on the adversarial (zero-sum) case, and while we only consider two agents, one or both agents may have arbitrary convex action sets at each stage of the game, rather than only selecting a policy in an MDP.

As another example of the expressive power of CEFGs, we demonstrate how they can efficiently model outcome uncertainty in EFGs. In games with outcome uncertainty, a (joint) action results in a probability distribution over future states, rather than a deterministic single state as in standard EFGs. Standard EFGs can model outcome uncertainty through the explicit use of random nodes, but as we shall see this can lead to a prohibitive explosion in the size of the game tree. While we feel this example is both conceptually straightforward and a significant research result in its own right, CEFGs have many other promising applications, including providing a method for introducing a limited amount of partial observability into stochastic games or MDPs while still allowing for tractable solutions. See (McMahan 2006) for discussion of other applications.

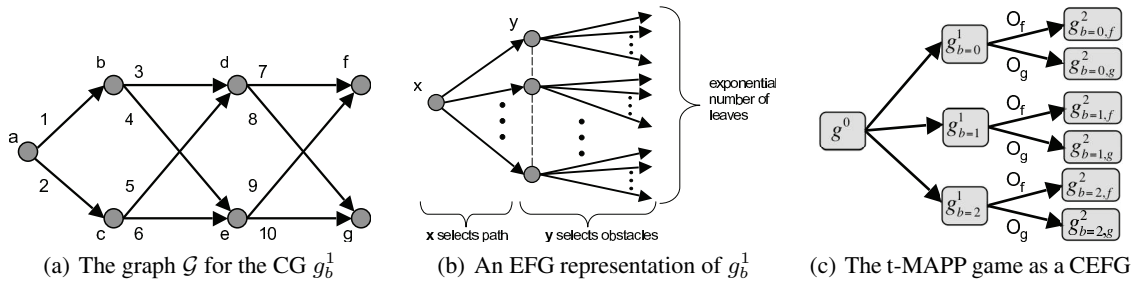


Figure 1: Representation of example games.

Bilinear-payoff convex games Bilinear-payoff convex games¹ (CGs) are a natural generalization of matrix games, though they have received surprisingly little attention in the literature since their introduction by Dresher & Karlin (1953). EFGs and (as we will show) CEFGs are special cases of convex games, as are other adversarial problems such as computing an optimal oblivious routing. For a general introduction to CGs and their application, see (McMahan 2006).

A two-player CG is defined by convex sets X and Y and a payoff matrix M_p for each player. The first player (say x) selects an action $x \in X$, the second (say y) simultaneously chooses $y \in Y$, and each player p receives a payoff $x^T M_p y$. In the zero-sum case, $M_y = -M_x \equiv M$ gives the amount x pays y , and an extension of the minimax theorem holds:

$$\min_{x \in X} \max_{y \in Y} x^T M y = \max_{y \in Y} \min_{x \in X} x^T M y.$$

This equilibrium is achieved for some $x^* \in X, y^* \in Y$. The convexity of X and Y implies that no explicit randomization is necessary; rather, a player can implicitly randomize over the corners of the action set by playing an interior point. Zero-sum EFGs can be represented as a CG and solved via the ellipsoid algorithm (Koller & Megiddo 1992); in fact, all zero-sum CGs can be solved in this manner. If X and Y are polyhedra (i.e., defined by a finite number of linear equality and inequality constraints), then in fact an equilibrium (x^*, y^*) can be found via linear programming (Koller, Megiddo, & von Stengel 1994).

Markov decision processes and CGs In this section we review some results on MDPs, and introduce an illustrative game that involves MDP planning. An MDP M is defined by a (finite) set of states S , a finite set of actions A , and transition dynamics defined by probabilities $\Pr(s' | s, a)$ for all $s, s' \in S$ and $a \in A$. We consider undiscounted MDPs with a fixed start state and a set of goal states. A cost vector c assigns a cost $c(s, a)$ to each state-action pair. For any finite set Q , let $\Delta(Q)$ be the $|Q|$ -dimensional probability simplex (that is, the set of distributions over Q). A stochastic policy is a function $\pi : S \rightarrow \Delta(A)$ that maps each state to a probability distribution over actions. The goal in standard MDP

planning is to find a policy π that minimizes the expected start-to-goal cost with respect to a fixed cost vector c .

It is well known that the set of stochastic policies for such an MDP can be represented via state-action visitation frequency vectors, and further that the set of these vectors is convex (for example, McMahan 2006). Here we demonstrate the result for a small example. The path-planning problem on the directed acyclic graph shown in Figure 1(a) can be interpreted as a deterministic MDP with $S = \{a, b, c, \dots, f, g\}$ and $A = \{1, 2, \dots, 10\}$, where a is the start state, f and g are the goals, and the actions available at a state are the directed edges out of that state. A state-action visitation vector for a policy π is a vector from \mathbb{R}^{10} indexed by the edges, which gives the probability² that a given edge is traversed under policy π . Thus the deterministic policy that chooses path (a, b, d, f) corresponds to the vector $x = (1, 0, 1, 0, 0, 0, 1, 0, 0, 0)$ because it uses edges 1, 3, and 7 only. The vector c assigns costs to edges, so the expected cost of this policy is exactly $x \cdot c$. The convex set of vectors x that correspond to valid stochastic policies can be defined via linear flow constraints. We have

$$\begin{aligned} x_1 + x_2 &= 1 && \text{(leave } a \text{ with prob 1)} \\ x_7 + x_8 + x_9 + x_{10} &= 1 && \text{(reach a goal with prob 1)} \end{aligned}$$

for the start and goal states, and each internal state has a flow constraint (e.g., $x_3 + x_4 = x_1$ for b) that the probability we enter the state equals the probability we leave. Combining these equality constraints with the constraint $x \geq 0$ defines the convex set X_G of valid edge (or state-action) visitation frequencies. We can recapture the stochastic policy π_x corresponding to a vector x by normalizing at each state; for example, the probability π_x takes edge 7 from d is $\frac{x_7}{x_7 + x_8}$. This convex representation of the set of stochastic policies for MDPs shows that planning in an MDP is a convex game: player x has strategy set X_G , y has the (trivial) action set $Y = \{c\}$, and $M = I$ (the identity matrix).

Now, we construct an illustrative CG where y has a more interesting set of actions: she chooses how to place at most B obstacles on edges of the graph, with each edge getting at most one obstacle. The cost on edge e is $c_e + 1$ if it contains an obstacle, and c_e otherwise (c gives the base costs of the edges, independent of the obstacles). Define the convex set

$$Y_B = \{y \mid y = c + z, z_i \geq 0, z_i \leq 1, \mathbf{1} \cdot z \leq B\}. \quad (1)$$

¹Our convex games are non-cooperative, and are unrelated to the super-modular coalitional games often called convex games in the cooperative game theory literature.

²If the graph was not acyclic, this would generalize to the expected number of visits to the edge.

The set of possible deterministic obstacle placements correspond to integer choices of z in Y_B . It can be shown that the corresponding y are exactly the corners of the set, and so an interior point of Y_B can be interpreted as a probability distribution over deterministic obstacle placements. We can now define a convex game where x chooses from X_G , y from Y_B , and $M = I$. Since y is a cost vector on edges and x is a frequency vector on edges, the dot product $x^T I y$ exactly calculates the expected value of the corresponding strategy/cost-vector pair. We call this game g_B^1 ; it is an adversarial-cost MDP, as introduced in (McMahan, Gordon, & Blum 2003).

We could have modeled this game as an exponentially-big (in the size of the graph) matrix game where the adversary y has combinatorially many pure strategies (deterministic obstacle placements), and the planner x has one pure strategy for each start-goal path in the graph (exponentially many in the size of the graph). The most concise EFG representation corresponds to the transformation of this matrix game to an EFG, as shown in Figure 1(b): first x selects a path x , and then y selects a placement of obstacles without knowledge of x (the dashed lines indicate that y 's decision nodes are in a single information set).

We now consider an extension to this game to help motivate CEFGs. Fix $B = 2$ for concreteness, and suppose that after playing g_B^1 , the players will then play another convex game g^2 . The exact second-stage game $g_{b,s}^2$ played depends on how many obstacles $b \in \{0, 1, 2\}$ player y placed in the first stage, and which goal state $s \in \{f, g\}$ player x reached, but the exact path taken by x and the exact placement of obstacles by y are irrelevant. For example, \mathcal{G} might be a subgraph of a larger graph that continues beyond f and g , and so $g_{b,s}^2$ is another path planning game on a different piece of the graph, where x starts from state s and y has $B - b$ obstacles available. Breaking the game into stages like this will allow y to potentially observe x 's location (and pick obstacles accordingly), or let x potentially observe y 's remaining budget. We call this game t-MAPP, because it can be viewed as a tiny instance of a multi-stage adversarial path planning (MAPP) problem. It is no longer clear that the t-MAPP game can be modeled as a convex game, as we now have significant temporal structure.

Consider representing t-MAPP as an EFG: the first stage of the game, g^1 , is modeled by Figure 1(b); for each of the exponentially many leaves of this game, we now attach a copy of the appropriate $g_{s,b}^2$ (also represented as an EFG) for that leaf. But $g_{s,b}^2$ could be an arbitrary convex game, so it is quite possible that each copy is itself exponentially large when represented as an EFG. So, clearly, t-MAPP is intractable when represented as a standard EFG. In the next section, we introduce CEFGs, which will give us a natural polynomial-sized representation of t-MAPP, of the form given in Figure 1(c): at node g^0 player y picks $b \in \{0, 1, 2\}$, the number of obstacles she will place in stage one. At the stage one nodes g_b^1 , y picks a cost vector y that respects the budget b , while x simultaneously picks a path x (without knowledge of b or y). The transition to the next stage game then depends only on the relevant portion of the outcome: which state x reached, and how much of the budget

y exhausted. Unlike the EFG representation, the CEFG can “forget” the exact path taken by x and the exact obstacle configuration chosen by y .

Sequential games like this can be used to model a powerful class of multi-stage adversarial path planning problems: We have a large graph, and x starts at a fixed node and must travel to a goal. However, the goal is not yet known, and may in fact be chosen randomly or by the adversary. So, in the first stage, player x travels to an intermediate node in the graph. Player y partially controls the costs, just as in the g^1 game. After the first stage, the adversary or nature chooses the location of the goal, some information about the goal is revealed to x , and some information about x 's location may be revealed to y . Another movement stage occurs, in which x picks a policy to another intermediate node, and y selects another cost vector. The game continues in stages until the goal is revealed and reached by x . CEFGs can model MAPP problems with many kinds of uncertainty, including outcome uncertainty, stochastic and adversarial control of costs, stochastic and adversarial control of the goal states, and partial observability (McMahan 2006).

Generalizing Extensive-Form Games

In this section, we introduce CEFGs for the two-player, zero-sum case. Complete proofs as well as additional examples, commentary, and references can be found in (McMahan 2006, Ch. 4). Comments specific to the example t-MAPP are set in brackets [...].

A CEFG is played on a directed, finite game tree $T = \langle V, E \rangle$ rooted at s^* . Define $\mathcal{E}(s) = ((s^1, s^2), (s^2, s^3), \dots, (s^{k-1}, s^k = s))$, the sequence of edges on the path to s . The game is played by a set $N = \{0, 1, 2\}$ of players, where the 0 player is an optional random player. We again let $x = 1$ be the min player and $y = 2$ be the max player. Each player is active (selects an action) on an arbitrary subset of the internal (non-leaf) nodes,³ $V_p \subseteq V$; these are player p 's decision nodes. Let $\mathcal{A}(s) = \{p \mid s \in V_p\}$, the set of active players at s . We require $|\mathcal{A}(s)| \geq 1$ for all internal nodes $s \in V$, and $|\mathcal{A}(s)| = 0$ for leaves. [For t-MAPP, the set $V = \{g^0, g_{b=0}^1, \dots, g_{b=2,g}^2\}$. Node $g^0 \in V_y$ because at this node y selects her budget, but $g^0 \notin V_x$. Both players are active at all the nodes g_b^1 .]

As in EFGs, the decision nodes V_p for each player are partitioned into information sets $u \in U_p$. When play reaches a node s in information set u , then player p observes that the game has reached u , but not the specific $s \in u$; that is, all $s, s' \in u$ are indistinguishable to p . For nodes s where p is not active ($s \notin V_p$), we define (for notational convenience) a special “non-information set” \diamond_p . In particular, $\diamond_p \notin U_p$ and player p never observes when she is in \diamond_p . For any node $s \in V_p$, let $\phi_p(s)$ be the (unique) $u \in U_p$ such that $s \in u$, and let $\phi_p(s) = \diamond_p$ for $s \notin V_p$. To simplify notation, when u is not otherwise specified it can be read as

³Allowing strict subsets of players to be active at each node requires some notational gymnastics; however, it is necessary to maintain a direct transformation from EFGs to CEFGs.

$\phi_p(s)$. For any node s and player p , let $\text{obs}_p(s)$ be the sequence of player p information sets on the path to s : $\text{obs}_p(s)$ has an entry u for each state s' with $\phi_p(s') \neq \diamond_p$ on the unique path to s . [In t-MAPP, each of the first four nodes is in its own information set for y . For x , $\phi_x(g^0) = \diamond_x$, and $u_x^1 = \{g_{b=0}^1, g_{b=1}^1, g_{b=2}^1\} \in U_x$ is an information set because x does not observe how many obstacles y decided to place.]

A few notes on notation: Each information set u we mention is associated with a player (say p), so for example x_u is an action taken by p at u . A bar indicates a tuple over players, e.g., \bar{x} is a joint action. Entries in a tuple over players are indexed with a subscript $p \in N$, $\bar{x} = \{x_0, x_1, x_2\}$. A bar over a capital symbol denotes a set of such tuples: \bar{X}_s is a set of joint actions.

All nodes in an information set u for p share the same set X_u of actions available to p ; this is natural because p cannot differentiate among these nodes. For this presentation, we assume the action set $X_u \subseteq \mathbb{R}^{n_u}$ is a bounded polyhedron (defined by a finite number of linear equalities and inequalities). At s , each player $p \in \mathcal{A}(s)$ selects an action $x_p \in X_u$. We denote by $\mathcal{C}(X_u)$ the finite set of corners (extreme points) of X_u . The set $\mathcal{C}(X_u)$ may be exponentially large even if X_u has a compact representation. We view the set $\mathcal{C}(X_u)$ as the primitive actions of the game (actions that are actually taken in the world), with the interior points interpreted as probability distributions over $\mathcal{C}(X_u)$. We define $X_{\diamond_p} = \{1\}$, and let $\bar{X}_s = \bigotimes_{p \in N} X_u$ (where \bigotimes is the Cartesian set product), so that a joint action $\bar{x} = (x_0, x_1, x_2) \in \bar{X}_s$ is a tuple over all the players, even though the players $p \notin \mathcal{A}(s)$ do not actually make a decision and have no immediate knowledge that s was reached. [In t-MAPP, the action set for y at g^0 is $\Delta(\{0, 1, 2\})$. Each g^1 node is in a different information set for y (she knows what b she picked), and has action set Y_b from Equation (1). For x , all the g^1 nodes are in the same information set, and the set of actions is simply the set X_G of possible stochastic paths.]

Payoffs are made at internal nodes, not just at leaves as for EFGs. The payoff from x to y at node s when x plays x and y plays y is given by the bilinear form $x^T M^s y$, specified by the matrix M^s . Since $X_{\diamond_p} = \{1\}$, we use this same notation to indicate payoffs where only one player selects an action, and at leaves where no actions are selected. [In t-MAPP, there are no payoffs at g^0 (M is the zero matrix), but at each node g^1 we have $M = I$, and so payoffs are the dot product between the cost vector y and edge-frequency vector x .]

We may have an exponential set of possible actions $\mathcal{C}(X_u)$, but we cannot afford that many nodes in T , let alone children of one node. Thus, each internal node $s \in V$ has a small (i.e., feasible to work with) set of successors, denoted $\text{succ}(s)$. The successor that occurs next in the game is chosen via a probability distribution that is a function of the joint action \bar{x} . For each $p \in \mathcal{A}(s)$ and $s' \in \text{succ}(s)$, the game specifies a linear function⁴ $f_p^{ss'} : X_u \rightarrow \mathbb{R}$; to avoid special cases define $f_p^{ss'}(x_p) = 1$ for $p \notin \mathcal{A}(s)$. The

⁴The linearity of the f -functions is necessary for the transformation from CEFGs to CGs for efficient solution.

probability that s' is the next node after s is given by:

$$\Pr(s' \mid s, \bar{x}) = \prod_{p \in N} f_p^{ss'}(x_p). \quad (2)$$

Thus, we require that these functions are chosen in such a way that $\Pr(s' \mid s, \bar{x})$ is always a well-defined probability distribution. The random player has one information set per node and $X_u = \{1\}$ for all $u \in U_0$. Thus, the random player is defined by constant f -functions, $f_0^{ss'}$. [In t-MAPP, the transition from stage 2 to stage 3 depends only on which goal x reaches, and so y 's f -functions all return 1. For x , we define

$$f_x^f = x_7 + x_9 \quad \text{and} \quad f_x^g = x_8 + x_{10}$$

where f^f is the f -function for g_b^1 to $g_{b,f}^2$ for $b \in \{0, 1, 2\}$, and similarly for f^g . Since $x \in X_G$ we have $x_7 + x_8 + x_9 + x_{10} = 1$ and $x_i \geq 0$, and so the products in Equation (2) produce a probability distribution.]

Two CEFGs G and G' are f -equivalent if they are identical except for their f functions, and if for all (s, s') , for all $\bar{x} \in \bar{X}_s$, $\Pr_G(s' \mid s, \bar{x}) = \Pr_{G'}(s' \mid s, \bar{x})$. Clearly G and G' are essentially the same game. We assume throughout that $f^{ss'}(x) \in [0, 1]$, without loss of generality (proof omitted). It is easy to show that for any G that does not satisfy this property, there is an f -equivalent G' that does.

The payoff to each player is simply the sum over the payoffs at each node visited in the game. A *complete history* h of a CEFG is the sequence of nodes and joint (primitive) actions that occurred in a play of the game; a history is composed of tuples $(s, (x, y))$ where s is the state and x and y are the actions selected by x and y respectively. The value of a history h is given by

$$V(h) = \sum_{(s, (x, y)) \in h} x^T M^s y.$$

Player x tries to minimize V , while player y tries to maximize it. Let \mathcal{H} be the set of all complete histories. A partial player history, h_p , is the (information set, action) sequence for player p so far in the game, ending with any player p information set. Let H_p be the set of all such histories for p .

A convex game can be immediately represented as a single-node CEFG, and in fact each node in a CEFG can be viewed as a convex game where the players' joint action determines not only the immediate bilinear payoff, but also a probability distribution over outcomes; information sets, however, imply that unlike in CGs, a player in a CEFG may not know his opponent's action set. Since MDPs are a special case of CGs, they can be modeled as a single-node CEFG.

Theory of CEFGs

In this section, we develop the concept of sufficient recall and show that for CEFGs with sufficient recall, a class of behavior strategies always contains an optimal policy. First, we establish some terminology regarding policy classes and probabilities.

Policy classes and probability The most general type of policy is a function from private randomness and $h_p \in H_p$ to X_u . When we refer to a general policy κ , it is from this class. We write $\bar{\kappa}_{-p}$ for a joint policy for all players except p , that is, $\bar{\kappa}_{-p} = (\kappa_1, \kappa_2, \dots, \kappa_{p-1}, \kappa_{p+1}, \dots, \kappa_n)$, and let $(\kappa_p, \bar{\kappa}_{-p})$ be the joint policy where players other than p play according to $\bar{\kappa}_{-p}$ and player p follows κ_p . We also consider the class of *implicit behavior reactive policies* (IBRPs). We say *reactive* because IBRPs are not history dependent: an IBRP β selects its action $\beta(u)$ as a function of only the current information set u . By *implicit behavior*, we mean β specifies an interior point of X_u , and then interprets that point as a probability distribution over $\mathcal{C}(X_u)$. Thus, β is a function from $u \in U_p$ to X_u .

Any joint policy induces a distribution on \mathcal{H} ; the probabilities we work with will be with respect to this distribution. To emphasize which joint policy is associated with a given probability or expectation, we include the policy as a condition, for example, $\Pr(s \mid \bar{\kappa})$. When s appears as an event, it is the subset of histories in \mathcal{H} in which s occurs. Similarly, the event u is the subset of histories where some $s \in u$ is reached, and (s, x_p) is the set of histories where player p selects action $x_p \in X_u$ from s . Once we fix $\bar{\kappa}$, V is a random variable, and we define $\mathcal{V}(\bar{\kappa}) = E[V \mid \bar{\kappa}]$, the expected payoff from x to y under joint policy $\bar{\kappa}$.

A policy κ_p for player p is *payoff equivalent* to another policy κ'_p , if for all $\bar{\kappa}_{-p}$ for the other players, $\mathcal{V}(\kappa_p, \bar{\kappa}_{-p}) = \mathcal{V}(\kappa'_p, \bar{\kappa}_{-p})$. In the rest of this section we define sufficient recall, and then show that for any policy κ in a sufficient recall CEF, there is a payoff equivalent IBRP. Sequence weights play a fundamental roll in this effort.

Sequence weights Sequence weights for CEFs are analogous to sequence weights in EFGs (Koller & Megiddo 1992; Koller, Megiddo, & von Stengel 1994) but do not by themselves contain enough information to represent a policy. (In the next section, we will augment the sequence weights with enough information to represent IBRPs.) Intuitively, the *sequence weight* $w_p(s \mid \kappa_p)$ is the probability player p reaches s by following κ_p , given that all other players (and their randomness) “conspire” to force p to s . More formally, define $\text{REL}(\kappa_p) \subseteq V$ to be the set of nodes potentially reachable (relevant) when κ_p is played, that is, $\text{REL}(\kappa_p) = \{s \mid \exists \bar{\kappa}_{-p} \text{ s.t. } \Pr(s \mid (\kappa_p, \bar{\kappa}_{-p})) > 0\}$. Define the sequence weight for s given κ_p by $w(s \mid \kappa_p) = 0$ if $s \notin \text{REL}(\kappa_p)$, and otherwise

$$w(s \mid \kappa_p) = \prod_{(t, t') \in \mathcal{E}(s)} \sum_{x \in \mathcal{C}(X_u)} \Pr((t, x) \mid t, \kappa_p) f_p^{tt'}(x),$$

Recall that we have $\Pr((s, 1) \mid s) = 1$ and $f_p^{ss'}(1) = 1$ when $p \notin \mathcal{A}(s)$, so the product is effectively over only those edges resulting from player p 's choices. The next lemma shows that the conditional probabilities in the definition are well defined.

Lemma 1. *For any p using policy κ_p and any two joint policies for the other players $\bar{\kappa}_{-p}$ and $\bar{\kappa}'_{-p}$, for any $s \in V_p$ where $\Pr(s \mid (\kappa_p, \bar{\kappa}_{-p})) > 0$ and $\Pr(s \mid (\kappa_p, \bar{\kappa}'_{-p})) > 0$, we have*

$$\Pr((s, x_p) \mid s, (\kappa_p, \bar{\kappa}_{-p})) = \Pr((s, x_p) \mid s, (\kappa_p, \bar{\kappa}'_{-p})).$$

In other words, given s is reached, the probability that κ_p plays x_p is independent of the opponents' joint policy. Intuitively this is the case because, as κ_p plays the game, its decisions can only depend on its own past actions and the information sets it observes, and conditioned on reaching s , these both must be independent of the other players' policies. Thus, as promised, when $s \in \text{REL}(\kappa_p)$, we can write $\Pr((s, x_p) \mid s, \kappa_p)$ for $\Pr((s, x_p) \mid s, (\kappa_p, \bar{\kappa}_{-p}))$.

A fundamental result is that if we know the sequence weights for each player, we can easily compute the probability that any state is reached:

Lemma 2. *For any $s \in V$ and any joint policy $\bar{\kappa}$,*

$$\Pr(s \mid \bar{\kappa}) = \prod_p w_p(s \mid \kappa_p).$$

For the next lemma, we define $E[x \mid s, \kappa_p] = \sum_{x \in \mathcal{C}(X_u)} \Pr((s, x) \mid s, \kappa_p) x$ when $s \in \text{REL}(\kappa_p)$.

Lemma 3. *If κ_p and κ'_p are two policies for player p such that*

$$E[x_p \mid s, \kappa_p] = E[x_p \mid s, \kappa'_p]$$

for all $s \in \text{REL}(\kappa_p) \cap \text{REL}(\kappa'_p)$, then $\text{REL}(\kappa_p) = \text{REL}(\kappa'_p)$, and further κ_p and κ'_p are payoff equivalent.

We will use this lemma to show that IBRPs are as powerful as general policies in sufficient recall CEFs.

Sufficient Recall A CEF has *sufficient recall* for player p if it has both: 1) *observation memory*: For all $u \in U_p$, and all $s, s' \in u$, $\text{obs}_p(s) = \text{obs}_p(s')$; and, 2) *sufficient action memory*: For any two policies κ_p and κ'_p and any joint policy $\bar{\kappa}_{-p}$ for the other players, for any $u \in U_p$ with $\Pr(u \mid (\kappa_p, \bar{\kappa}_{-p})) > 0$ and $\Pr(u \mid (\kappa'_p, \bar{\kappa}_{-p})) > 0$, and any $s \in u$, then $\Pr(s \mid u, (\kappa_p, \bar{\kappa}_{-p})) = \Pr(s \mid u, (\kappa'_p, \bar{\kappa}_{-p}))$.

Observation memory implies that player p 's information sets form a forest, and so knowing the current information set uniquely specifies the history of information sets (observations) that have previously occurred; hence player p has no incentive to remember the information sets visited. Sufficient action memory implies that given the observation of u , knowledge of the policy that has been followed so far provides no information about the actual $s \in u$. Thus, p has no incentive to remember the policy followed so far. Informally, if the game has sufficient recall for player p , then player p should be able to play optimally by selecting an action purely as a (random) function of the current information set. In order to prove this, we will need an alternative characterization of sufficient recall, which will allow us to prove some important structural properties of sequence weights.

Sufficient recall arises naturally in many CEFs; two general cases appear in this paper. The first case includes games where the action impacts the immediate payoff (say, the exact path chosen in an MDP), but only some finite information (say, the goal state reached) matters to the rest of the game. The second case occurs when the action chosen induces a probability distribution over outcomes, but only the actual outcome chosen by nature from this distribution matters to the rest of the game; we will see this is the case for EFGs with outcome uncertainty modelled as CEFs.

In an EFG, all nodes in an information set u have the same out-degree d , and each outgoing edge for a $s \in u$ is labeled with one of d outcome (or choice) labels. The action set of the EFG is the set of these labels. We can view the outcome labels as partitioning all of the edges out of u into d equivalence classes. We now define a generalization of this partition for CEFs via an equivalence relation \sim_p on pairs of edges out of u . For any two edges (s, s') and (t, t') out of u , $(s, s') \sim_p (t, t')$ if and only if there exists a constant $\alpha \geq 0$ such that for all $x \in X_u$, $f_p^{ss'}(x) = \alpha f_p^{tt'}(x)$. Let O_u be the set of equivalence classes at u defined by \sim_p . [In t-MAPP, all stage 2 nodes for x are in the single information set u_x^1 . Recall that the f -functions are independent of b , and so we have the partition of the six edges out of u_x^1 into two sets, those leading to f (labeled o_f in Figure 1(c)), and those leading to g (labeled o_g).]

We have the following key Lemma:

Lemma 4. *For any CEF G , there exists an f -equivalent CEF G' such that if $(s, s') \sim_p (t, t')$ in G , then for all $x \in X_u$, in G' we have $f_p^{ss'}(x) = f_p^{tt'}(x)$.*

CEFs that satisfy the condition of Lemma (4) are called f -canonical, and for the remainder of this paper, we assume (without loss of generality) all CEFs considered are f -canonical. The transformation to an f -canonical representation is essential—our solution technique is not even defined on CEFs that do not have this property. Under this assumption, we write $f_p^{u,o}$ for the f function shared by all edges out of u in outcome partition $o \in O_u$.

The player p sequence $\sigma_p(s)$ is the list of player p (information sets, outcome) tuples on the unique path to s ; it does not include tuples for states where $\phi_p(s) = \diamond_p$. A CEF has *sequence recall* for player p , if for all $u \in U_p$ and all $s, s' \in u$, $\sigma_p(s) = \sigma_p(s')$. [It is straightforward to verify that the game t-MAPP satisfies sequence recall: the key is the identical f -functions out of the g^1 nodes for x .]

Lemma 5. *In a f -canonical CEF with sequence recall, for any policy κ_p for player p , and any $s, s' \in u$, then $w(s | \kappa_p) = w(s' | \kappa_p)$, and when $w(s | \kappa_p) > 0$, for all $x \in X_u$, $\Pr((s, x) | s, \kappa_p) = \Pr((s', x) | s', \kappa_p)$.*

The intuition for the proof is that because the sequence to each $s \in u$ is the same, there is no way for the policy κ_p to differentiate the nodes s and s' (even using knowledge of past actions and observations), and hence, it must play identically at both.

Lemma (5) reveals significant tree-like structure of sequence weights in CEFs with sequence recall: we can now think of sequence weights being associated with each player's information set tree, rather than with the overall game tree. We extend some notation to account for this: we write $w(u | \kappa_p)$ for the unique value $w(s | \kappa_p)$ shared by all $s \in u$. Each non-root information set u_2 for player p has a unique (information set, outcome) parent, which we identify by $\text{upred}_p(u_2) = (u_1, o_1)$. If u is a root information set, we write $\text{upred}_p(u) = \emptyset$. Any state s occurring after some player p information set (that is, with a non-empty $\sigma_p(s)$) has a unique (information set, outcome) predecessor, namely the last tuple in $\sigma_p(s)$. We write $\text{upred}(s_2) = (u_1, o_1)$ to

identify such a parent. Any state s occurring before any player p information set has $w(s | \kappa_p) = 1$. All immediate successor states of u reached via an edge in a fixed outcome partition o must have the same sequence weight; we write $w(u, o | \kappa_p)$ for this value. In summary, for any node $s \in u_2$ where $(u_1, o_1) = \text{upred}_p(u_2)$, we write any of the following equivalently:

$$w_p(s | \kappa_p) = w_p(u_2 | \kappa_p) = w_p(u_1, o_1 | \kappa_p).$$

Theorem 6. *A f -canonical CEF has sufficient recall for player p if and only if it has sequence recall for player p .*

Showing sequence recall implies sufficient recall is fairly straightforward; the other direction is more difficult. It relies on an alternative characterization of sufficient action memory, and then the construction of a suitable contradiction. Based on this theorem, we can apply the results and notation for sequence recall games to sufficient recall games; this will be critical for the construction of an equivalent convex game in the next section.

Now we can give this section's principal result: IBRPs are as powerful as arbitrary policies in sufficient recall CEFs. The theoretical work done earlier in this section makes this result straightforward.

Theorem 7. *For sufficient-recall CEFs, for any policy κ_p for player p , there exists a payoff equivalent implicit behavior reactive policy.*

Proof. Let κ_p be an arbitrary policy for p . A consequence of Lemma (5) is that for all $s, s' \in u$, when $s, s' \in \text{REL}(\kappa_p)$,

$$E[x_p | s, \kappa_p] = E[x_p | s', \kappa_p].$$

Call this value x_u for each u where it is defined, and pick x_u arbitrarily in X_u for the remaining $u \in U_p$. Then, we define an implicit behavior policy β_κ by $\beta_\kappa(u) = x_u$. By Lemma (3) β_κ and κ_p are payoff equivalent. \square

From CEFs to CGs

Theorem 7 shows that when playing sufficient recall CEFs, it suffices to consider only IBRPs. Now, we show that for each player the set of IBRPs can be represented as a convex set \mathcal{W} , and the value of the game is bilinear in this representation. Thus, we can solve zero-sum sufficient-recall CEFs using linear programming on the convex game defined by the sets \mathcal{W} and corresponding bilinear objective function.

To differentiate players x and y , we use u and X_u to denote player x 's information and action sets, and similarly v and Y_v for y . The random player only affects the game through her sequence weights, which we write as $w_0(s)$.

An IBRP for player x can be viewed as a vector from the convex set

$$\tilde{X} = \bigotimes_{u \in U_x} X_u.$$

The set \tilde{X} is a Cartesian product of convex sets, and so it is also a convex set. Define \tilde{Y} analogously for y , and let $\beta_x \in \tilde{X}$ and $\beta_y \in \tilde{Y}$ be two IBRPs. Let $u = \phi_x(s)$ and $v = \phi_y(s)$, $x = \beta_x(u)$, and $y = \beta_y(v)$, and define

$$\begin{aligned} \mathcal{V}(s) &= \Pr(s | (\beta_x, \beta_y)) x^T M^s y \\ &= w_0(s) w(s | \beta_x) w(s | \beta_y) x^T M^s y \end{aligned}$$

using Lemma (2). Then the expected total payoff from x to y is

$$\mathcal{V} = \sum_{s \in \text{REL}(\beta_x, \beta_y)} \mathcal{V}(s).$$

Unfortunately, $\mathcal{V}(s)$ is not bilinear in β_x and β_y . We now develop an alternative convex representation for IBRPs in which $\mathcal{V}(s)$ is bilinear. Our use of sequence weights as variables is analogous to the technique in (Koller, Megiddo, & von Stengel 1994), but our approach must also represent the implicit behavior taken at each X_u , as this is not defined by the sequence weights alone.

We represent an IBRP for x as an element of a set \mathcal{W}_x (and analogously for y with a set \mathcal{W}_y). Our construction of \mathcal{W}_x relies on the sets

$$X_u^c = \{(\alpha x, \alpha) \mid x \in X_u, \alpha \geq 0\} \subseteq \mathbb{R}^{n_u+1}$$

for each $u \in U_x$. The set X_u^c is the *cone extension* of X_u , and it is also convex; in fact, if X_u is a polyhedron, then so is X_u^c . Define

$$\tilde{X}^c = \bigotimes_{u \in U_x} X_u^c.$$

We will have $\mathcal{W}_x \subseteq \tilde{X}^c$. We work with a vector $\omega_x \in \tilde{X}^c$ by writing $\omega_x = \langle (x_u^c, w_u) \mid u \in U_x \rangle$, where $(x_u^c, w_u) \in X_u^c$, and so $x_u^c \in \mathbb{R}^{n_u}$ and $w_u \in \mathbb{R}$ are defined for all $u \in U_x$ by ω_x . The set \mathcal{W}_x is defined by the following constraints:

$$\begin{aligned} (x_u^c, w_u) &\in X_u^c \\ w_u &= 1 && \forall u \in U_x \text{ with } \text{upred}_p(u) = \emptyset \\ w_u &= f_x^{u', o'}(x_{u'}^c) && \forall u \in U_x \text{ with } \text{upred}_p(u) = (u', o'). \end{aligned}$$

The set \mathcal{W}_x is convex as \tilde{X}^c is convex and the constraints are linear (recall the f -functions are linear).

We associate each $\omega_x \in \mathcal{W}_x$ with a behavior policy via the function g defined by $g(\omega_x) = \beta_{\omega_x}$ where $\beta_{\omega_x} \in \tilde{X}$ is the IBRP defined by

$$\beta_{\omega_x}(u) = \begin{cases} (1/w_u)x_u^c & \text{when } w_u > 0 \\ \text{arbitrary} & \text{otherwise.} \end{cases}$$

It can be shown that $w_u = w(u \mid \beta_{\omega_x})$, and so the w_u variables are in fact sequence weights. The function g is not quite a bijection, because β_{ω_x} can be set arbitrarily at information sets that are never reached; however, it is easy to show that optimizing over the set \mathcal{W}_x is equivalent to optimizing over \tilde{X} . With these tools, we can prove that payoffs are bilinear in the \mathcal{W}_x representation:

Theorem 8. *In a two-player, zero-sum, sufficient recall CEFGs, represent x 's IBRPs as \mathcal{W}_x , and player y 's IBRPs as \mathcal{W}_y . Then, for any $\omega_x \in \mathcal{W}_x$ and $\omega_y \in \mathcal{W}_y$, the payoff $\mathcal{V}(g(\omega_x), g(\omega_y))$ is a bilinear function of ω_x and ω_y .*

An immediate corollary is that zero-sum sufficient-recall CEFGs can be solved in polynomial time using the linear programming solution for the convex game with strategy sets \mathcal{W}_x and \mathcal{W}_y and the payoff matrix implied by Theorem (8).

Modeling Outcome Uncertainty in EFGs

We now show that the class of extensive-form games with outcome uncertainty (which we term *perturbed games*) can be compactly represented as CEFGs, while their EFG representations are exponentially larger. This lets us generalize extensive-form games in much the same way that MDPs generalize deterministic path planning. The analogy is not perfect, because perturbed games are still representable as EFGs (but at the cost of an exponential blowup in size), while a general MDP cannot be modeled by any deterministic planning problem.

Fix a standard (unperturbed) EFG G , and let O_u be the set of outcomes (labels) at an information set u , so O_u is in a 1-1 correspondence with the children of s for all $s \in u$. In G , O_u is exactly the set of actions available to the player at u : the player chooses some action $o \in O_u$, and the game state transitions deterministically to the appropriate successor node associated with the choice o .

A perturbed EFG introduces a level of indirection between a player's action selection and the outcome of the action by decoupling the set of actions available from the set of outcomes O_u . A perturbed EFG (G, A) is specified as a standard EFG G together with a perturbation model A . The perturbation model specifies a finite set $A_u = \{p_1, p_2, \dots\}$ of probabilistic (meta-)actions for each player at each information set. Each action p_i specifies a distribution over possible outcomes, so $p_i \in \Delta(O_u)$. Hence the analogy to MDPs, where an action at a state is defined by the distribution over successor states it induces. The perturbed game (G, A) is played as follows: when $p \in A_u$ is selected by the player active at u , the actual outcome $o \in O_u$ is sampled according to the distribution p , and the game transitions to the unique successor of the current $s \in u$ that corresponds to outcome o , as if the player had selected o in the unperturbed game. It is standard to assume that the player at u observes which outcome o actually occurred; other players in the game only observe this if they observed the player's action at u in the unperturbed game.

A perturbed EFG (G, A) can be represented as an extensive-form game G_A . For each information set u in G , for each $s \in u$, we introduce a new random node for each $p \in A_u$ in order to model the outcome uncertainty associated with meta-action p . Thus, s is given successor random nodes in 1-1 correspondence with A_u , and each of these random nodes has successors in 1-1 correspondence with O_u (and hence the successors of s in G). Applying this transformation at a single information set u blows up the size of the game representation by a factor of $\mathcal{O}(|A_u|)$. Introducing a perturbation model at each information set will increase the size of the game tree exponentially: while (G, A) can be represented in space $\mathcal{O}(|G| + |A|)$, the EFG representation G_A can take space $\mathcal{O}(|G| \cdot |A_u|^d)$, where d is the depth of the original game tree.

An example of this transformation at a single node s_1 is shown in parts (a) and (b) of Figure 2. Part (b) of the figure shows the introduction of random nodes r_1 and r_2 that implement the perturbation model. The game tree of (b) thus remembers both which action outcome the player wanted to happen (a versus b) as well as which action outcome actu-

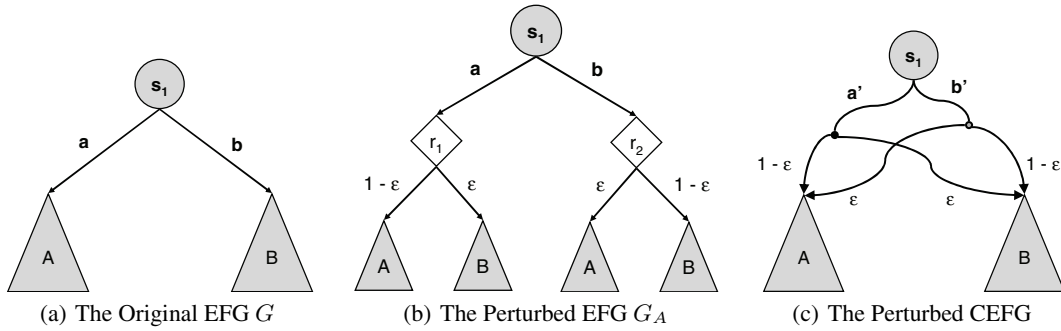


Figure 2: Representing a perturbed game as an EFG and as a CEFG. The state s_1 is in a singleton information set for simplicity. We have $O_u = \{A, B\}$, where the triangles labeled with these outcomes correspond to the rest of the game tree, which could be very large. The perturbation model is $A_u = \{a, b\}$ where the meta-action a achieves outcome A with probability $1 - \epsilon$ and outcome B with probability ϵ for some small value ϵ , and similarly for b .

ally happened: hence there are two copies of the subtrees A and B , doubling the size of the EFG.

We now show how to transform an EFG with a perturbation model into a compact sufficient-recall CEFG. To represent an *unperturbed* EFG as a CEFG, we keep the same game tree, and replace the finite action set O_u with the convex action set $X_u = \Delta(O_u)$ at each information set u . To represent the *perturbed* game (G, A) we keep the game tree of G , but the set of available actions X_u at u will be the convex hull of the probability distributions in A_u . Each $x \in X_u$ then corresponds to a valid probability distribution over O_u achievable by playing a mixture of the actions A_u . It is straightforward to verify that this CEFG satisfies sufficient recall and has a representation of size $\mathcal{O}(|G| + |A|)$. A schematic for this representation is shown in part (c) of Figure 2. The key is that the game tree remains the same as G , and the space of possible outcome distributions is stored independently via the sets X_u .

The concise representation perturbed EFGs gives a simple polynomial-time algorithm for finding approximate trembling-hand equilibria (also called perfect equilibria) for extensive-form games: namely, one simply solves the CEFG version of the original EFG where on each action the player gets a random action with probability ϵ instead of the one chosen. Solving for perfect equilibria (or some other form of sequential equilibria) can be critical in practice, but only very recently have algorithms for finding such equilibria been investigated (Miltersen & Sorensen 2006).

We have modeled outcome uncertainty efficiently using CEFGs, but have not fully tapped the class’s representational power. In particular, we have not used the ability to model both players simultaneously playing at a single node, and we have not used the ability to model different numbers of outcomes at different states in the same information set. Both of these abilities enable exponentially smaller representations for some EFGs.

Conclusion and Future Directions

We have introduced CEFGs and shown that they can be solved efficiently in the zero-sum sufficient-recall case.

CEFGs can compactly represent some games whose EFG representation is exponentially large, thus allowing for polynomial-time solution of games that before were intractable. Two general example domains were presented: an extension of MDPs to an adversarial setting with temporal structure and sequential observation, and the extension of EFGs to allow for actions with stochastic outcomes. It remains a promising line of future research to investigate other applications of this expressive framework.

Acknowledgments The authors thank Avrim Blum for useful discussions and suggesting the line of inquiry that eventually led to this work. This research was supported in part by a grant from DARPA’s CS2P program.

References

- Dresher, M., and Karlin, S. 1953. Solutions of convex games as fixed-points. In Kuhn, H. W., and Tucker, A., eds., *Contributions To The Theory of Games: Volume 2*, number 28 in Annals of Mathematics Studies. Princeton University Press. 75–86.
- Koller, D., and Megiddo, N. 1992. The complexity of two-person zero-sum games in extensive form. *Games and Economic Behavior* 4(4):528–552.
- Koller, D.; Megiddo, N.; and von Stengel, B. 1994. Fast algorithms for finding randomized strategies in game trees. In *Proceedings of the 26th Annual ACM Symposium on the Theory of Computing*.
- McMahan, H. B.; Gordon, G. J.; and Blum, A. 2003. Planning in the presence of cost functions controlled by an adversary. In *ICML 2003*.
- McMahan, H. B. 2006. *Robust Planning in Domains with Stochastic Outcomes, Adversaries, and Partial Observability*. Ph.D. Dissertation, Carnegie Mellon University. Tech report CMU-CS-06-166.
- Miltersen, P. B., and Sorensen, T. B. 2006. Computing sequential equilibria for two-player games. In *SODA ’06*.
- Petrik, M., and Zilberstein, S. 2007. Average-reward decentralized markov decision processes. In *IJCAI 2007*.