Dynamic Thresholding and Pruning for Regret Minimization

Noam Brown, Christian Kroer, and Tuomas Sandholm

Carnegie Mellon University, Pittsburgh PA 15213, USA,
noamb@cmu.edu,ckroer@cs.cmu.edu,sandholm@cs.cmu.edu

Abstract. Regret minimization is widely in determining strategies for imperfect-information games and in online learning. In large games, each iteration of the algorithm may be prohibitively slow. For this reason, pruning – in which parts of the decision tree are not traversed in every iteration – has emerged as an essential method for dealing with large games. The ability to prune is a primary reason why the Counterfactual Regret Minimization (CFR) algorithm using regret matching has emerged as the most popular iterative algorithm for imperfect-information games, despite its relatively poor convergence bound. In this paper, we introduce dynamic thresholding, in which a threshold is set at every iteration such that any action in the decision tree with probability below the threshold is set to zero probability. This enables pruning for the first time in a wide range of algorithms. We prove that dynamic thresholding can be applied to Hedge while increasing its convergence bound by only a constant factor in terms of number of iterations. Experiments demonstrate a substantial improvement in performance relative to the number of nodes touched.

1 Introduction

We introduce *dynamic thresholding* for online learning algorithms, in which a threshold is set at every iteration such that any action with probability below the threshold is set to zero probability. This enables pruning for the first time in a wide range of algorithms. The theory that we derive applies to each of the two central goals in the area:

1. Regret minimization in any setting, where there can be any number of players in a general-sum game, and our agent may not even know what the game is (except that the agent knows the available actions when it is her turn to move).
2. Converging to an ϵ-Nash equilibrium in a two-player zero-sum game. Results for (1) immediately imply results for this setting by having our algorithm be used by both agents.

We will introduce this first for the application of solving zero-sum imperfect-information games (that is, games like heads-up poker), and then explain how the results directly carry over to non-zero-sum games and to general regret minimization. Furthermore, the results apply to both extensive-form and normal-form representations.

Imperfect-information extensive-form games are a way to model strategic multi-step interactions between players that have hidden information, such as negotiations, auctions, cybersecurity settings, and medical settings. A Nash equilibrium in relatively small two-player zero-sum games containing around 10^8 nodes can be found precisely using a linear program [13]. For larger games, iterative algorithms are used to converge to an ϵ-Nash equilibrium. There are a number of such iterative algorithms, the most popular of which is *Counterfactual Regret Minimization (CFR)* [26]. CFR minimizes regret independently at each decision point (called an information set) in the game tree using any regret-minimizing algorithm. By far the most popular regret-minimizing algorithm to use within CFR is *regret matching (RM)* and variants of RM [15, 12, 11, 5]. CFR+, a variant of CFR with RM, was recently used to essentially solve Limit Texas Hold'em, the largest imperfect-information game ever to be essentially solved and the first that is played competitively by humans [2, 24]. That game (after lossless abstraction [13] as a preprocessor) has over 10^{13} information sets.

When computing strategies for large imperfect-information games, repeatedly traversing the entire game tree with an iterative algorithm may be prohibitively slow. For this reason, pruning—in which parts of the game tree are not traversed on every iteration—has emerged as an essential method for dealing with large games. The ability to prune is a primary reason why the *Counterfactual Regret Minimization algorithm (CFR)* that uses *Regret Matching (RM)* at each information set is the most popular algorithm for imperfect-information games, despite its relatively poor $O(\sqrt{|A|T})$ cumulative regret.

While regret-minimizing algorithms other than RM can be used within CFR, and iterative algorithms other than CFR exist with better convergence bounds in terms of the number of iterations needed [16, 22, 14], CFR with RM exhibits superior empirical performance in large games [17]. A primary reason for this is that CFR with RM is able to put zero probability on some actions, and therefore prune large sections of the game tree, particularly in large games. That is, it need not traverse the entire game tree on each iteration. This behavior is shared by some other regret minimizing algorithms, but is relatively uncommon and is considered a desirable property [20]. The ability to prune enables each iteration to be completed far more quickly. While the benefit of pruning varies depending on the game, it can easily be multiple orders of magnitude even in small games [18, 4]. Moreover, the benefits of pruning typically grow with the size of the game.

In this paper we introduce *dynamic thresholding* that allows pruning to be applied in a wider range of algorithms, and applied more frequently in settings that already support pruning. We focus on *Hedge* [10, 19], also known as the *exponentially-weighted forecaster*, which is the most popular regret-minimizing algorithm in domains other than extensive-form game solving, on RM, and on the *Excessive Gap Technique* (EGT) [21, 14], which converges to an ϵ-Nash equilibrium in two-player zero-sum games in $O(\frac{1}{\epsilon})$, that is, in significantly fewer iterations than CFR which converges in $O(\frac{1}{\epsilon^2})$.

Dynamic thresholding sets a minimum probability threshold on each iteration, and any action with probability below that threshold is set to zero probability. We decrease the threshold over time, where the decrease is asymptotically slower than the possible decrease of action probabilities in Hedge and EGT. Thus, poor actions may eventually be played with probability below the threshold, and those paths in the game tree can then be pruned using the same methods as are used in CFR with RM (which we will describe in detail later in the paper). We prove that dynamic thresholding increases the convergence bound in Hedge and RM by only a small constant factor, where the factor depends on how aggressively the threshold is set. This holds whether Hedge and RM are used as stand-alone algorithms in any setting, or as subroutines within CFR for game-tree settings.

The remainder of this paper is structured as follows. In the next section, we cover background on imperfect-information extensive-form games, Nash equilibria, and CFR. Then, we formally introduce dynamic thresholding in CFR with Hedge/RM and prove its convergence guarantees. Then, we present experimental results that show that dynamic thresholding leads to a dramatic improvement in the performance of CFR with Hedge and of EGT. Finally, we will conclude and discuss other potential future uses of dynamic thresholding.

2 Background

In this section we present the background needed for the rest of the paper. The first subsection introduces the standard notation. The subsection after that covers CFR, explained in a more general way than usual because we want to also consider other regret matching algorithms within CFR than the usual, which is RM. Finally, the third subsection presents the pruning variants that have been introduced for CFR.

2.1 Notation

In an imperfect-information extensive-form game there is a finite set of players, $\mathcal{P}$. H is the set of all possible histories (nodes) in the game tree, represented as a sequence of actions, and includes the empty history. $A(h)$ is the actions available in a history and $P(h) \in \mathcal{P} \cup c$ is the player who acts at that history, where c denotes chance. Chance plays an action $a \in A(h)$ with a fixed probability $\sigma_c(h, a)$ that is known to all players. The history h' reached after an action is taken in h is a child of h, represented by $h \cdot a = h'$, while h is the parent of h'. More generally, h is an ancestor of h' (and h' is a descendant of h), represented by $h \sqsubset h'$, if there exists a sequence of actions from h to h'. $Z \subseteq H$ are terminal histories for which no actions are available. For each player $i \in \mathcal{P}$, there is a payoff function $u_i : Z \to \Re$. If $P = \{1, 2\}$ and $u_1 = -u_2$, the game is two-player zero-sum. We define $\varDelta_i = \max_{z \in Z} u_i(z) - \min_{z \in Z} u_i(z)$ and $\varDelta = \max_i \varDelta_i$.

Imperfect information is represented by *information sets* for each player $i \in \mathcal{P}$ by a partition $\mathcal{I}_i$ of $h \in H : P(h) = i$. For any information set $I \in \mathcal{I}_i$, all histories $h, h' \in I$ are indistinguishable to player i, so $A(h) = A(h')$. $I(h)$ is the information set I where $h \in I$. $P(I)$ is the player i such that $I \in \mathcal{I}_i$. $A(I)$ is the set of actions such that for all $h \in I$, $A(I) = A(h)$. $|A_i| = \max_{I \in \mathcal{I}_i} |A(I)|$ and $|A| = \max_i |A_i|$. We define $U(I)$ to be the maximum payoff reachable from a history in I, and $L(I)$ to be the minimum. That is, $U(I) = \max_{z \in Z, h \in I : h \sqsubseteq z} u_{P(I)}(z)$ and $L(I) = \min_{z \in Z, h \in I : h \sqsubseteq z} u_{P(I)}(z)$. We define $\varDelta(I) = U(I) - L(I)$ to be the range of payoffs reachable from a history in I. We similarly define $U(I, a)$, $L(I, a)$, and $\varDelta(I, a)$ as the maximum, minimum, and range of payoffs (respectively) reachable from a history in I after taking action a. We define $D(I, a)$ to be the set of information sets reachable by player $P(I)$ after taking action a. Formally, $I' \in D(I, a)$ if for some history $h \in I$ and $h' \in I'$, $h \cdot a \sqsubseteq h'$ and $P(I) = P(I')$.

A strategy $\sigma_i(I)$ is a probability vector over $A(I)$ for player i in information set I. The probability of a particular action a is denoted by $\sigma_i(I, a)$. Since all histories in an information set belonging to player i are indistinguishable, the strategies in each of them must be identical. That is, for all $h \in I$, $\sigma_i(h) = \sigma_i(I)$ and $\sigma_i(h, a) = \sigma_i(I, a)$. We define σ_i to be a probability vector for player i over all available strategies $\varSigma_i$ in the game. A strategy profile σ is a tuple of strategies, one for each player. $u_i(\sigma_i, \sigma_{-i})$ is the expected payoff for player i if all players play according to the strategy profile $\langle \sigma_i, \sigma_{-i} \rangle$. If a series of strategies are played over T iterations, then $\bar{\sigma}_i^T = \frac{\sum_{t \in T} \sigma_i^t}{T}$.

$\pi^\sigma(h) = \varPi_{h' \rightarrow a \sqsubseteq h} \sigma_{P(h)}(h, a)$ is the joint probability of reaching h if all players play according to σ. $\pi_i^\sigma(h)$ is the contribution of player i to this probability (that is, the probability of reaching h if all players other than i, and chance, always chose actions leading to h). $\pi_{-i}^\sigma(h)$ is the contribution of all players other than i, and chance. $\pi^\sigma(h, h')$ is the probability of reaching h' given that h has been reached, and 0 if $h \not\sqsubseteq h'$. In a *perfect-recall* game, $\forall h, h' \in I \in \mathcal{I}_i$, $\pi_i(h) = \pi_i(h')$. In this paper we focus on perfect-recall games. Therefore, for $i = P(I)$ we define $\pi_i(I) = \pi_i(h)$ for $h \in I$. We define the average strategy $\bar{\sigma}_i^T(I)$ for an information set I to be

$$\bar{\sigma}_i^T(I) = \frac{\sum_{t \in T} \pi_i^{\sigma_i^t} \sigma_i^t(I)}{\sum_{t \in T} \pi_i^{\sigma^t}(I)} \tag{1}$$

A *best response* to σ_{-i} is a strategy σ_i^* such that $u_i(\sigma_i^*, \sigma_{-i}) = \max_{\sigma_i' \in \varSigma_i} u_i(\sigma_i', \sigma_{-i})$. A *Nash equilibrium*, is a strategy profile where every player plays a best response. Formally, a Nash equilibrium is a strategy profile σ^* such that $\forall i$, $u_i(\sigma_i^*, \sigma_{-i}^*) = \max_{\sigma_i' \in \varSigma_i} u_i(\sigma_i', \sigma_{-i}^*)$. We define a *Nash equilibrium strategy* for player i as a strategy σ_i that is part of any Nash equilibrium. In two-player zero-sum games, if σ_i and σ_{-i} are both Nash equilibrium strategies, then $\langle \sigma_i, \sigma_{-i} \rangle$ is a Nash equilibrium. We define an *ϵ-equilibrium* as a strategy profile σ^* such that $\forall i$, $u_i(\sigma_i^*, \sigma_{-i}^*) + \epsilon \geq \max_{\sigma_i' \in \varSigma_i} u_i(\sigma_i', \sigma_{-i}^*)$.

2.2 Counterfactual Regret Minimization (CFR)

Counterfactual Regret Minimization (CFR) is the most popular algorithm for extensive-form imperfect-information games. In CFR, the strategy vector for each information set is determined according to a regret-minimization algorithm [26]. Typically, *regret matching (RM)* is used as the regret-minimization algorithm in CFR even though Hedge has a better convergence bound (in terms of the number of iterations) [6]. One reason is that the vanilla version of Hedge does not support pruning of any paths in extensive-form games because all probabilities in Hedge are strictly positive. In section 3 we introduce a modification to Hedge that allows pruning, so we cover both Hedge and RM in this section.

Our analysis of CFR makes frequent use of *counterfactual value*. Informally, this is the expected utility of an information set given that player i tries to reach it. For player i at information set I given a strategy profile σ, this is defined as

$$v^{\sigma}(I) = \sum_{h \in I} \Big(\pi^{\sigma}_{-i}(h) \sum_{z \in Z} \big(\pi^{\sigma}(h, z) u_i(z) \big) \Big) \tag{2}$$

The counterfactual value of an action a is

$$v^{\sigma}_i(I, a) = \sum_{h \in I} \Big(\pi^{\sigma}_{-i}(h) \sum_{z \in Z} \big(\pi^{\sigma}(h \cdot a, z) u_i(z) \big) \Big) \tag{3}$$

Let σ^t be the strategy profile used on iteration t. The *instantaneous regret* on iteration t for action a in information set I is

$$r^t(I, a) = v^{\sigma^t}_{P(I)}(I, a) - v^{\sigma^t}_{P(I)}(I) \tag{4}$$

and the *regret* for action a in I on iteration T is

$$R^T(I, a) = \sum_{t \in T} r^t(I, a) \tag{5}$$

Additionally, $R^T_+(I, a) = \max\{R^T(I, a), 0\}$ and $R^T(I) = \max_a\{R^T_+(I, a)\}$. Regret for player i in the entire game is

$$R^T_i = \max_{\sigma'_i \in \Sigma_i} \sum_{t \in T} \big(u_i(\sigma'_i, \sigma^t_{-i}) - u_i(\sigma^t_i, \sigma^t_{-i}) \big) \tag{6}$$

In regret matching, a player picks a distribution over actions in an information set in proportion to the positive regret on those actions. Formally, on each iteration $T + 1$, player i selects actions $a \in A(I)$ according to probabilities

$$\sigma^{T+1}(I, a) = \begin{cases} \frac{R^T_+(I,a)}{\sum_{a' \in A(I)} R^T_+(I,a')}, & \text{if } \sum_{a'} R^T_+(I, a') > 0 \\ \frac{1}{|A(I)|}, & \text{otherwise} \end{cases} \tag{7}$$

If a player plays according to regret matching in information set I on every iteration then on iteration T, $R^T(I) \leq \Delta(I)\sqrt{|A(I)|}\sqrt{T}$ [6].

In Hedge, a player picks a distribution over actions in an information set according to

$$\sigma^{T+1}(I,a) = \frac{e^{\eta_T R^T(I,a)}}{\sum_{a' \in A(I)} e^{\eta_T R^T(I,a')}} \tag{8}$$

where η_T is a tuning parameter. There is a substantial literature on how to set η_T for best performance [6, 7]. If a player plays according to Hedge in information set I on every iteration t and uses $\eta_t = \sqrt{\frac{2\ln(|A(I)|)}{T}}$ then on iteration T, $R^T(I) \leq \varDelta(I)\sqrt{2\ln(|A(I)|)T}$ [6].

If a player plays according to CFR on every iteration then

$$R_i^T \leq \sum_{I \in \mathcal{I}_i} R^T(I) \tag{9}$$

So, as $T \to \infty$, $\frac{R_i^T}{T} \to 0$.

In two-player zero-sum games, if both players' average regret satisfies $\frac{R_i^T}{T} \leq \epsilon$, their average strategies $\langle \bar{\sigma}_1^T, \bar{\sigma}_2^T \rangle$ form a 2ϵ-equilibrium [25]. Thus, CFR constitutes an anytime algorithm for finding an ϵ-Nash equilibrium in zero-sum games.

2.3 Pruning Techniques

In this section we discuss pruning techniques that allow parts of the game tree to be skipped within CFR iterations.

(Partial) Pruning Typically, regret is updated by traversing each node in the game tree separately for each player, and calculating the contribution of a history $h \in I$ to $r^t(I,a)$ for each action $a \in A(I)$. If a history h is reached in which $\pi^{\sigma^t}_{-i}(h) = 0$ (that is, an opponent's reach is zero), then from (2) and (3) the strategy at h contributes nothing on iteration t to the regret of $I(h)$ (or to the information sets above it). Moreover, any history that would be reached beyond h would also contribute nothing to its information set's regret because $\pi^{\sigma^t}_{-i}(h') = 0$ for every history h' where $h \sqsubset h'$ and $P(h') = P(h)$. Thus, when traversing the game tree for player i, there is no need to traverse beyond any history h when $\pi^{\sigma^t}_{-i}(h) = 0$. The benefit of this form of pruning, which we refer to as *partial pruning*, varies depending on the game, but empirical results show a factor of 30 improvement in some small games [18].

Regret-Based Pruning (RBP) While partial pruning allows one to prune paths that an *opponent* reaches with zero probability, the recently introduced *regret-based pruning* (RBP) algorithm allows one to also prune paths that the *traverser* reaches with zero probability [4]. However, this pruning is necessarily temporary. Consider an action $a \in A(I)$ such that $\sigma^t(I,a) = 0$, and assume for now that it is known action a will not be played with positive probability until some far-future iteration t' (in RM, this would be the case if $R^t(I,a) \ll 0$). Since action a is played with zero probability on iteration t, the strategy played and reward received following action a (that is, in $D(I,a)$) will not contributed to the regret for any information set preceding action a on iteration t. In fact, what happens in $D(I,a)$ has no bearing on the rest of the game tree until iteration t' is reached. So one can "procrastinate" until iteration t' in deciding what happened beyond action a on iteration t, $t+1$, ..., $t'-1$.

Upon reaching iteration t', rather than individually making up the $t'-t$ iterations over $D(I,a)$, one can instead do a *single* iteration, playing against the *average* of the opponents' strategies in the $t'-t$ iterations that were missed, and declare that strategy was played on all the $t'-t$ iterations. This accomplishes the work of the $t'-t$ iterations in a single traversal. Moreover, since player i never plays action a with positive probability between iterations t and $t'-1$, that means every *other* player can apply partial pruning on that part of the game tree for the $t'-t$ iterations, and skip it completely. This, in turn, means that player i has free rein to play whatever she wants in $D(I,a)$ without affecting the regrets of the other players. In light of that, and of the fact that player i gets to decide what is played in $D(I,a)$ after knowing what the other players have played, player i might as well play a strategy that ensures zero regret for all information sets $I' \in D(I,a)$ in the iterations t to $t'-1$. For instance, player i can play a best response to the opponents' average strategy from iterations t to $t'-1$; this is what we do in the experiments in this paper.

Regret-based pruning only allows a player to skip traversing $D(I,a)$ for as long as $\sigma^t(I,a)=0$. Thus, in RM if $R^{t_0}(I,a)<0$ we can prune the game tree beyond action a from iteration t_0 onward in consecutive iterations as long as for the current iteration t_1 we have

$$\sum_{t=1}^{t_0} v^{\sigma^t}(I,a) + \sum_{t=t_0+1}^{t_1} \pi^{\sigma^t}_{-i}(I)U(I,a) \leq \sum_{t=1}^{t_1} v^{\sigma^t}(I) \tag{10}$$

Once this no longer holds, skipping ceases. If we later find another t_0 that satisfies $R^{t_0}(I,a)<0$, we do another sequence of iterations where we skip traversing after a, and so on.

3 Dynamic Thresholding

The pruning methods described in Section 2.3 can only be applied when an action is played with zero probability. This makes pruning incompatible with Hedge, because in Hedge all the action probabilities are strictly positive. This motivates our introduction of *dynamic thresholding*, in which low-probability actions are set to zero probability.

In dynamic thresholding for Hedge, we set any action with probability less than $\frac{(C-1)\sqrt{\ln(|A(I)|)}}{\sqrt{2}|A(I)|^2\sqrt{t}}$ on iteration t (where $C \geq 1$) to zero probability and normalize the remaining action probabilities accordingly so they sum to 1. If an action is thresholded, this deviation from what Hedge calls for may lead to worse performance and therefore higher regret. However, using the threshold that we just specified above, we ensure that the new regret is within a constant factor C of the traditional regret bound.

Theorem 1. *If player $P(I)$ plays according to Hedge in an information set I for T iterations using threshold $\frac{(C-1)\sqrt{\ln(|A(I)|)}}{\sqrt{2}|A(I)|^2\sqrt{t}}$ with $C \geq 1$ on every iteration t, then $R^T(I) \leq C\sqrt{2}\Delta(I)\sqrt{\ln(|A(I)|)}\sqrt{T}$.*

To apply the above theorem within CFR, we get from Equation 9 that one can then just sum the regrets of all information sets I to bound the total regret for this player.

Dynamic thresholding can in general be applied to any regret minimization algorithm. We present Theorem 1 specifically for Hedge in order to tailor the threshold for that algorithm, which provides a tighter theoretical bound. In Theorem 2, we also show that dynamic thresholding can be applied to RM. However, it results in very little, if any, additional pruning. This is because RM is very unlikely in practice to put extremely small probabilities on actions. Nevertheless, we prove that dynamic thresholding applies to RM for the sake of completeness and for its potential theoretical applications. Note that the formula for the threshold is now different.

Theorem 2. *If player $P(I)$ plays according to regret matching in an information set I for T iterations using threshold $\frac{C^2-1}{2C|A(I)|^2\sqrt{t}}$ with $C \geq 1$ on every iteration t, then $R^T(I) \leq C\Delta(I)\sqrt{|A(I)|}\sqrt{T}$.*

Again, to apply the above theorem within CFR, we get from Equation 9 that one can then just sum the regrets of all information sets I to bound the total regret for this player.

4 Regret-Based Pruning for Hedge

In this section we describe how dynamic thresholding enables regret-based pruning when using Hedge. To use RBP, it is necessary to determine a lower bound on the number of iterations for which an action will have zero probability. In RM without dynamic thresholding this is simply the minimum number of iterations it would take an action to achieve positive regret, as shown in (10). In Hedge with dynamic thresholding, we instead must determine the minimum number of iterations it would take for an action to reach probability above the dynamic threshold.

Let $R^{T_0}(I,a)$ be the regret for an action a in information set I on iteration T_0. If $\sigma^{T_0}(I,a) < \frac{(C-1)\sqrt{\ln(|A(I)|)}}{\sqrt{2}|A(I)|^2\sqrt{T_0}}$, where $\sigma^{T_0}(I,a)$ is defined according to (8), then pruning can begin on iteration T_0. By Theorem 1, we can prune the game tree following action a on any consecutive iteration T after that if

$$\frac{e^{\eta_T\left(R^{T_0}(I,a)+U(I,a)(T-T_0)\right)}}{\sum_{a' \in A(I)} e^{\eta_T\left(R^T(I,a')+\sum_{T'=T_0+1}^{T} v^{T'}(I,a')\right)}} < \frac{(C-1)\sqrt{\ln(|A(I)|)}}{\sqrt{2}|A(I)|^2\sqrt{t}} \tag{11}$$

Once this no longer holds, skipping ceases. If we later find another T_0 that satisfies the condition above, we do another sequence of iterations where we skip traversing after a, etc.

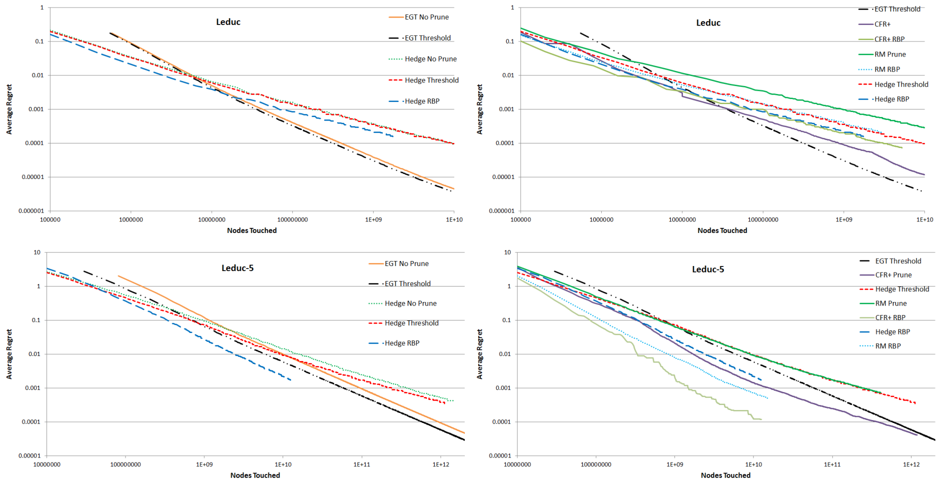

Fig. 1. Performance of EGT, CFR with Hedge, and CFR with RM on Leduc and Leduc-5. CFR with Hedge is shown without any pruning (vanilla Hedge), with dynamic thresholding, and with RBP. EGT is shown without any pruning (vanilla EGT) and with dynamic thresholding. CFR with RM is shown with partial pruning (vanilla RM) and with RBP. Dynamic thresholding on RM resulted in identical performance to vanilla RM, and is therefore not shown separately.

5 Experiments

We tested dynamic thresholding with and without RBP on a standard benchmark game called Leduc Hold'em [23] and an enlarged variant of Leduc Hold'em featuring more actions, called Leduc-5. Leduc Hold'em is a popular benchmark for imperfect-information game solving due to its feasible size and strategic complexity. In Leduc Hold'em, there is a deck consisting of six cards: two each of Jack, Queen, and King. There are two rounds. In the first round, each player places an ante of 1 chip in the pot and receives a single private card. A round of betting then takes place with a two-bet maximum, with Player 1 going first. A public shared card is then dealt face up and another round of betting takes place. Again, Player 1 goes first, and there is a two-bet maximum. If one of the players has a pair with the public card, that player wins. Otherwise, the player with the higher card wins. In standard Leduc Hold'em, all bets in the first round are 1 chip, while all bets in the second round are 2 chips. In Leduc-5, there are

5 bet sizes to choose from: in the first round the betting options are 1, 2, 4, 8, or 16 chips, while in the second round the betting options are 2, 4, 8, 16, or 32 chips. Leduc Hold'em contains 288 information sets, compared to $34,224$ for Leduc-5.

Hedge requires the user to set the tuning parameter η_t. When proving worst-case regret bounds, the parameter is usually defined as a function of $\Delta(I)$ for an information set I (for example, $\eta_t = \frac{\sqrt{8\ln(|A(I)|)}}{\Delta(I)\sqrt{t}}$) [6]. However, this is overly pessimistic in practice, and better performance can be achieved with heuristics while still guaranteeing convergence, albeit at a weaker convergence bound.[1] In our experiments, we set $\eta_t = \frac{\sqrt{\ln(|A(I)|)}}{3\sqrt{\mathrm{VAR}(I)_t}\sqrt{t}}$, where $\mathrm{VAR}(I)_t$ is the observed variance of $v(I)$ up to iteration t, based on a heuristic by Chaudhuri et al. [8].

In addition to the regret-minimization algorithms which are the main focus of this paper, for comparison we also experimented with the leading gradient-based algorithm for finding ϵ-equilibrium in zero-sum games, the *excessive gap technique (EGT)* [16, 21]. It converges to an ϵ-equilibrium in two-player zero-sum games in $O(\frac{1}{\epsilon})$ iterations, that is, in significantly fewer iterations than CFR which converges in $O(\frac{1}{\epsilon^2})$. In this EGT variant the gradient is computed by traversing the game tree (see also [17]). This enables pruning and dynamic thresholding to be implemented in EGT as well.

Figure 1 shows the performance of dynamic thresholding on Hedge and EGT against the vanilla versions of the algorithm as well as against the benchmark algorithms CFR+ and CFR with RM.[2] The two figures on the left show that dynamic thresholding benefits EGT and Hedge, and the relative benefit increases with game size. In Leduc-5, dynamic thresholding improves the performance of EGT by a factor of 2, and dynamic thresholding combined with RBP improves the performance of CFR with Hedge by a factor of 7. The graphs on the right show that, when using thresholding and RBP, Hedge outperforms RM in Leduc, but RM outperforms Hedge in Leduc-5. RM's better performance in Leduc-5 is due to more widespread pruning than Hedge.

While CFR+ exhibits the best performance in Leduc-5, there are several drawbacks to the algorithm that cause it not to be usable in all settings. First, CFR+ does not converge in theory when combined with RBP. The noisy performance of CFR+ with RBP in Leduc-5, and the weaker performance of CFR+ with RBP when compared with vanilla CFR+ in Leduc, are consequences of this. Second, while CFR+ in practice outperforms CFR with RM or Hedge, it has a

[1] Convergence is still guaranteed so long as $\Delta(I)$ is replaced with a value that has a constant lower and upper bound, though the worst-case bound may be worse.

[2] We present our results with the number of nodes touched on the x axis. Nodes touched is a hardware- and implementation-independent proxy for time. Hedge involves exponentiation when determining strategies, which takes longer than the simple floating point operations of RM. In our implementation, regret matching traverses 36% more nodes per second than Hedge. However, in large-scale multi-core implementations of CFR, memory access is the bottleneck on performance and therefore the penalty for using Hedge should not be as significant.

worse theoretical bound. Moreover, in the long run EGT appears to outperform CFR+. Finally, CFR+ is not compatible with sampling, which is commonly used in large imperfect-information games.

Figure 2 shows the performance of EGT and Hedge with different aggressiveness of dynamic thresholding. For EGT, we threshold by $\frac{c}{T}$, where the number shown in the legend is c. For Hedge, we threshold by $\frac{d\sqrt{\ln(|A|)}}{\sqrt{2}|A|^2\sqrt{T}}$, where d is shown in the legend (all of those values satisfy the theory of this paper). The results show that the performance is not sensitive to the parameter.

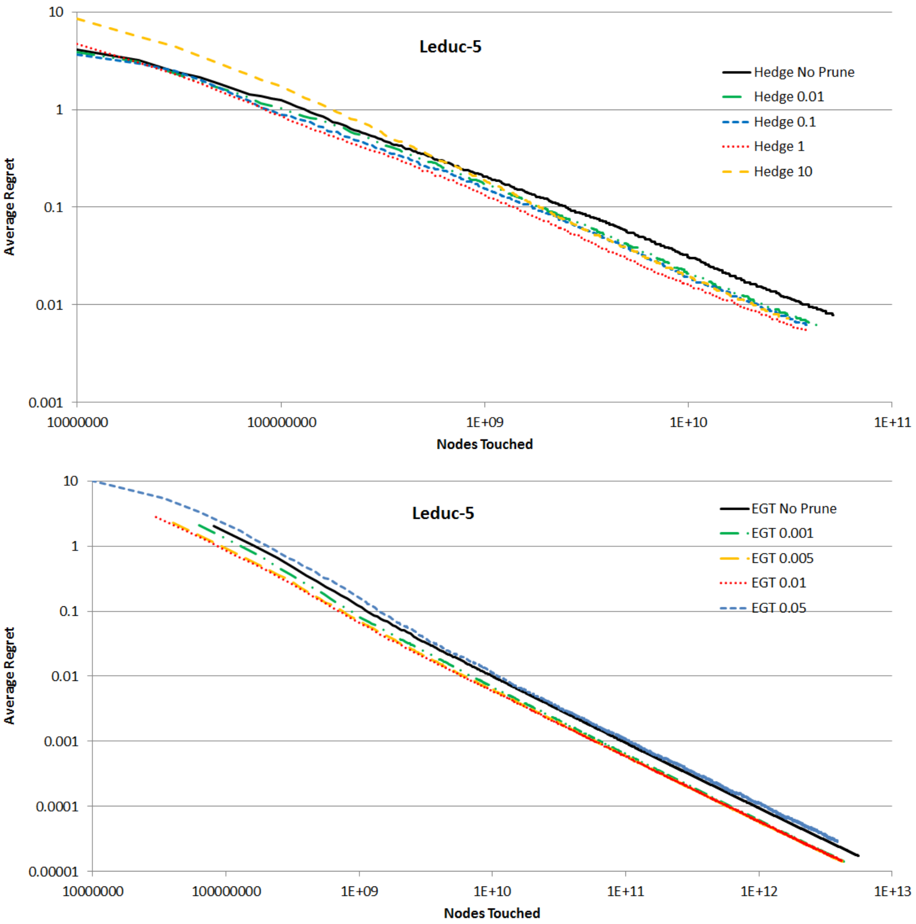

Fig. 2. Varying the aggressiveness of dynamic thresholding.

6 Conclusion and Future Research

We introduced *dynamic thresholding* for online learning algorithms, in which a threshold is set at every iteration such that any action with probability below the threshold is set to zero probability. This enables pruning for the first time in a wide range of algorithms. We showed that it can be applied to both Regret Matching and Hedge—regardless of whether they are used in isolation for any problem or as subroutines at each information set within Counterfactual Regret Minimization, the most popular algorithm for solving imperfect-information game trees. We proved that the regret bound increases by only a small constant factor, and each iteration becomes faster due to enhanced pruning. Our experiments demonstrated substantial speed improvements in Hedge; the relative speedup increases with problem size.

We also developed a version of the leading gradient-based algorithm for solving imperfect-information games, the excessive gap technique, where we compute the gradient based on a traversal of the game tree, thereby enabling the use of dynamic thresholding and pruning. Experiments again showed that they lead to a significant speedup, and that the relative speedup increases with problem size.

Our results on Hedge might also be useful for boosting when Hedge is used therein [10, 20]. The idea is that low-weight weak learners and/or low-weight training instances (as an analogy to low-probability actions in our paper) would then not need to be run, which may lead to significant time savings.

Future work also includes studying whether the idea of dynamic thresholding could be applied to other iterative algorithms that place at least some small positive probability on all actions (e.g., [22, 9]).

Bibliography

[1] Blackwell, David. An analog of the minmax theorem for vector payoffs. *Pacific Journal of Mathematics*, 6:1–8, 1956.

[2] Bowling, Michael, Burch, Neil, Johanson, Michael, and Tammelin, Oskari. Heads-up limit hold'em poker is solved. *Science*, 347(6218):145–149, January 2015.

[3] Brown, Noam and Sandholm, Tuomas. Regret transfer and parameter optimization. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2014.

[4] Brown, Noam and Sandholm, Tuomas. Regret-based pruning in extensive-form games. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, 2015.

[5] Brown, Noam, Ganzfried, Sam, and Sandholm, Tuomas. Hierarchical abstraction, distributed equilibrium computation, and post-processing, with application to a champion no-limit Texas Hold'em agent. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2015.

[6] Cesa-Bianchi, Nicolo and Lugosi, Gabor. *Prediction, learning, and games*. Cambridge University Press, 2006.

[7] Cesa-Bianchi, Nicolo, Mansour, Yishay, and Stoltz, Gilles. Improved second-order bounds for prediction with expert advice. *Machine Learning*, 66(2-3): 321–352, 2007.

[8] Chaudhuri, Kamalika, Freund, Yoav, and Hsu, Daniel J. A parameter-free hedging algorithm. In *Advances in neural information processing systems*, pp. 297–305, 2009.

[9] Daskalakis, Constantinos, Deckelbaum, Alan, and Kim, Anthony. Near-optimal no-regret algorithms for zero-sum games. *Games and Economic Behavior*, 92:327–348, 2015.

[10] Freund, Yoav and Schapire, Robert. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[11] Gibson, Richard. *Regret Minimization in Games and the Development of Champion Multiplayer Computer Poker-Playing Agents*. PhD thesis, University of Alberta, 2014.

[12] Gibson, Richard, Lanctot, Marc, Burch, Neil, Szafron, Duane, and Bowling, Michael. Generalized sampling and variance in counterfactual regret minimization. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2012.

[13] Gilpin, Andrew and Sandholm, Tuomas. Lossless abstraction of imperfect information games. *Journal of the ACM*, 54(5), 2007.

[14] Gilpin, Andrew, Peña, Javier, and Sandholm, Tuomas. First-order algorithm with $\mathcal{O}(\ln(1/\epsilon))$ convergence for ϵ-equilibrium in two-person zero-sum games. *Mathematical Programming*, 133(1–2):279–298, 2012. Conference version appeared in AAAI-08.

[15] Hart, Sergiu and Mas-Colell, Andreu. A simple adaptive procedure leading to correlated equilibrium. *Econometrica*, 68:1127–1150, 2000.

[16] Hoda, Samid, Gilpin, Andrew, Peña, Javier, and Sandholm, Tuomas. Smoothing techniques for computing Nash equilibria of sequential games. *Mathematics of Operations Research*, 35(2):494–512, 2010. Conference version appeared in WINE-07.

[17] Kroer, Christian, Waugh, Kevin, Kılınç-Karzan, Fatma, and Sandholm, Tuomas. Faster first-order methods for extensive-form game solving. In *Proceedings of the ACM Conference on Economics and Computation (EC)*, 2015.

[18] Lanctot, Marc, Waugh, Kevin, Zinkevich, Martin, and Bowling, Michael. Monte Carlo sampling for regret minimization in extensive games. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, pp. 1078–1086, 2009.

[19] Littlestone, Nick and Warmuth, M. K. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994.

[20] Luo, Haipeng and Schapire, Robert E. A drifting-games analysis for online learning and applications to boosting. In *Advances in Neural Information Processing Systems*, pp. 1368–1376, 2014.

[21] Nesterov, Yurii. Excessive gap technique in nonsmooth convex minimization. *SIAM Journal of Optimization*, 16(1):235–249, 2005.

[22] Pays, François. An interior point approach to large games of incomplete information. In *AAAI Computer Poker Workshop*, 2014.

[23] Southey, Finnegan, Bowling, Michael, Larson, Bryce, Piccione, Carmelo, Burch, Neil, Billings, Darse, and Rayner, Chris. Bayes' bluff: Opponent modelling in poker. In *Proceedings of the 21st Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 550–558, July 2005.

[24] Tammelin, Oskari, Burch, Neil, Johanson, Michael, and Bowling, Michael. Solving heads-up limit texas hold'em. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, 2015.

[25] Waugh, Kevin, Schnizlein, David, Bowling, Michael, and Szafron, Duane. Abstraction pathologies in extensive games. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2009.

[26] Zinkevich, Martin, Bowling, Michael, Johanson, Michael, and Piccione, Carmelo. Regret minimization in games with incomplete information. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, 2007.

Appendix

A Proof of Theorem 1

Proof. We use $\eta = \frac{\sqrt{2\ln(|A(I)|)}}{\Delta(I)\sqrt{T}}$ and define $\Phi(R^t(I))$ as

$$\Phi(R^t(I)) = \frac{1}{\eta}\ln\Big(\sum_{a\in A(I)} e^{\eta R^t(I,a)}\Big) \tag{12}$$

Since for all $a \in A(I)$ we know $e^{\eta R^t(I,a)} > 0$, so

$$\max_{a\in A(I)} R^T(I,a) \le \Phi(R^T(I)) \tag{13}$$

We prove inductively on t that

$$\Phi(R^t(I)) \le \frac{\ln(|A(I)|)}{\eta} + C(\Delta(I))^2\eta t \tag{14}$$

If (14) holds for all t, then from (13) the lemma is satisfied.

For $t = 1$, dynamic thresholding produces the same strategy as vanilla Hedge, so (14) is trivially true. We now assume that (14) is true for $t-1$ and consider iteration $t > 1$. Vanilla Hedge calls for a probability vector $\sigma^t(I)$ that, if played on every iteration t, would result in (14) holding for T. Dynamic thresholding creates a new strategy vector $\hat{\sigma}^t(I)$. Let $\delta^t(a) = \hat{\sigma}^t(I,a) - \sigma^t(I,a)$ and $\delta^t = \max_{a\in A(I)} \delta^t(a)$.

In the worst case, all but one action is reduced to zero and the probability mass is added to the single remaining action. Thus, $|\delta^t(a)| \le \frac{(C-1)\sqrt{\ln(|A(I)|)}}{\sqrt{2}|A(I)|\sqrt{t}}$. After playing $\hat{\sigma}^t(I,a)$ on iteration t, we have

$$\Phi(R^t(I)) \le \frac{1}{\eta}\ln\Big(\sum_{a\in A(I)} e^{\eta\left(R^{t-1}+r^t(I,a)\right)}\Big)$$

$$\Phi(R^t(I)) \le \frac{1}{\eta}\ln\Big(\sum_{a\in A(I)} e^{\eta\left(R^{t-1}+v^t(I,a)-v^t(I)\right)}\Big)$$

$$\Phi(R^t(I)) \qquad \le \qquad \frac{1}{\eta}\ln\Big(\sum_{a\in A(I)} e^{\eta\left(R^{t-1}+v^t(I,a)-\sum_{a'\in A(I)}\left(\hat{\sigma}^t(I,a')v^t(I,a')\right)\right)}\Big)$$

Since $\hat{\sigma}^t(I,a') = \sigma^t(I,a) + \delta^t(a)$, we get

$$\Phi(R^t(I)) \le \frac{1}{\eta}\ln\Big(\sum_{a\in A(I)} e^{\eta\left(R^{t-1}+v^t(I,a)-\sum_{a'\in A(I)}\left(\sigma^t(I,a')v^t(I,a')+\delta(a')v^t(I,a')\right)\right)}\Big)$$

Since $v^t(I,a') \le \Delta(I)$ and $\delta^t(a') \le \delta^t$, this becomes

$$\Phi(R^t(I)) \le \frac{1}{\eta}\ln\Big(e^{\eta\delta^t\Delta(I)|A(I)|}\sum_{a\in A(I)} e^{\eta\Big(R^{t-1}+v^t(I,a)-\sum_{a'\in A(I)}\big(\sigma^t(I,a')v^t(I,a')\big)\Big)}\Big)$$

$$\Phi(R^t(I)) \le \delta^t\Delta(I)|A(I)|+\frac{1}{\eta}\ln\Big(\sum_{a\in A(I)} e^{\eta\Big(R^{t-1}+v^t(I,a)-\sum_{a'\in A(I)}\big(\sigma^t(I,a')v^t(I,a')\big)\Big)}\Big)$$

Since $v^t(I,a)-\sum_{a'\in A(I)}\big(\sigma^t(I,a')v^t(I,a')$ is the original update Hedge intended, we apply Theorem 2.1 from [6] and Lemma 1 from [3] to get

$$\Phi(R^t(I)) \le \delta^t\Delta(I)|A(I)| + \Phi(R^{t-1}(I)) + \frac{(\Delta(I))^2\eta}{2}$$

Since $\delta^t < \frac{(C-1)\Delta(I)\eta}{2|A(I)|}$, we get

$$\Phi(R^t(I)) \le \Phi(R^{t-1}(I)) + C(\Delta(I))^2\eta$$

Substituting the bound on $\Phi(R^{t-1}(I))$ we arrive at

$$\Phi(R^t(I)) \le \frac{\ln(|A(I)|)}{\eta} + C(\Delta(I))^2\eta t$$

This satisfies the inductive step.

B Proof of Theorem 2

Proof. We find it useful to define

$$\Phi(R^T(I)) = \sum_{a\in A(I)}\big(R^T(I,a)^2_+\big) \tag{15}$$

We prove inductively on T that

$$\Phi(R^T(I)) \le C^2\big(\Delta(I)\big)^2 A(I)T \tag{16}$$

If (16) holds, then $R(I) \le C\Delta(I)\sqrt{|A(I)|}\sqrt{T}$. On iteration 1, regret matching calls for probability $\frac{1}{|A(I)|}$ on each action, which is above the threshold. Thus, dynamic thresholding produces identical strategies as vanilla regret matching, so from Theorem 2.1 in [6], (16) holds.

We now assume (16) holds for iteration $T-1$ and consider iteration $T > 1$. Vanilla regret matching calls for a probability vector $\sigma^T(I)$ that, if played, would result in (16) holding for T. Dynamic thresholding creates a new strategy vector $\hat{\sigma}^T(I)$ in which $\hat{\sigma}^T(I,a) = 0$ if $\sigma^T(I,a) \le \frac{C^2-1}{2C|A(I)|^2\sqrt{T}}$. After reducing actions to zero probability, the strategy vector is renormalized. Let $\delta(a) = \hat{\sigma}^T(I,a) - \sigma^T(I,a)$ and $\delta = \max_{a \in A(I)} \delta(a)$. In the worst case, all but one action is reduced to zero and the probability mass is added to the single remaining action. Thus, $|\delta(a)| \leq \frac{C^2-1}{2C|A(I)|\sqrt{T}}$.

After playing $\hat{\sigma}^T(I,a)$ on iteration T, we have

$$\Phi(R^T(I)) \leq \sum_{a \in A(I)} \big(R^{T-1}(I,a) + r^T(I,a)\big)^2_+$$

From Lemma 7 in [18], we get

$$\Phi(R^T(I)) \quad \leq \quad \Big(\Phi(R^{T-1}(I)) \;+\; 2\sum_a (R_+^{T-1}(I,a) r^T(I,a)) \;+\; r^T(I,a)^2\Big)$$

$$\Phi(R^T(I)) \quad \leq \quad \Big(\Phi(R^{T-1}(I)) \;+\; 2\sum_a (R_+^{T-1}(I,a) r^T(I,a)) \;+\; (\Delta(I))^2\Big)$$

From (4) and (7),

$$r^T(I,a) = v^T(I,a) - \sum_{a' \in A(I)} \big(\hat{\sigma}^T(I,a') v^T(I,a')\big)$$

Since $\hat{\sigma}^T(I,a) = \sigma(I,a)^T + \delta(a)$, we get

$$r^T(I,a) = v^T(I,a) - \sum_{a' \in A(I)} \Big(\big(\sigma^T(I,a') + \delta(a)\big) v^T(I,a')\Big)$$

Regret matching satisfies the Blackwell condition [1] which, as shown in Lemma 2.1 in [6], means

$$\sum_{a \in A(I)} \Big((R_+^{T-1}(I,a)\big(v^T(I,a) \quad - \quad \sum_{a' \in A(I)} \big(\sigma^T(I,a') v^T(I,a')\big)\big)\Big) \qquad \leq \qquad 0$$

Thus, we are left with

$$\sum_{a \in A(I)} (R_+^{T-1}(I,a) r^T(I,a)) \quad \leq \quad |\delta| \sum_{a \in A(I)} \Big(R_+^{T-1}(I,a) \sum_{a' \in A(I)} \big(v^T(I,a')\big)\Big)$$

Since $v^T(I,a') \leq \Delta(I)$, this leads to

$$\Phi(R^T(I)) \leq \Big(\Phi(R^{T-1}(I)) + 2|\delta| \sum_{a \in A(I)} \big(R_+^{T-1}(I,a)\Delta(I)|A(I)|\big) + (\Delta(I))^2|A(I)|\Big)$$

By the induction assumption,

$$\sum_{a \in A(I)} \big(R_+^{T-1}(I,a)\big)^2 \leq C^2(\Delta(I))^2|A(I)|(T-1)$$

so by Lemma 5 in [18]

$$\sum_{a \in A(I)} R_+^{T-1}(I,a) \leq C\Delta(I)|A(I)|\sqrt{T-1}$$

This gives us

$$\Phi(R^T(I)) \quad \leq \quad \Big(\Phi(R^{T-1}(I)) \ + \ (\Delta(I))^2|A(I)|\big(2C|\delta||A(I)|\sqrt{T-1} \ + \ 1\big)\Big)$$

Since $|\delta| < \frac{C^2-1}{2C|A(I)|\sqrt{T}}$ this becomes

$$\Phi(R^T(I)) \leq \Big(\Phi(R^{T-1}(I)) + C^2(\Delta(I))^2|A(I)|\Big)$$

Substituting the bound of $\Phi(R^T(I))$ we get

$$\Phi(R^T(I)) \leq C^2(\Delta(I))^2|A(I)|T$$

This satisfies the inductive step.