

# Iterated Regret Minimization in Game Graphs

Emmanuel Filiot    Tristan Le Gall    Jean-François Raskin  
efiliot@ulb.ac.be    tlegall@ulb.ac.be    jraskin@ulb.ac.be  
Université Libre de Bruxelles

**Abstract.** Iterated regret minimization has been introduced recently by J.Y. Halpern and R. Pass in classical strategic games. For many games of interest, this new solution concept provides solutions that are judged more reasonable than solutions offered by traditional game concepts – such as *Nash equilibrium* –. In this paper, we investigate iterated regret minimization for infinite duration two-player quantitative non-zero sum games played on graphs.

## 1 Introduction

The analysis of complex interactive systems like embedded systems or distributed systems is a major challenge of computer aided verification. Zero-sum games on graphs provide a good framework to model interactions between a component and an environment as they are strictly competitive. However in the context of modern interactive systems, several components may interact and be controlled independently. Non-zero sum games on graphs are more accurate to model such systems, as the objectives of the components are not necessarily antagonist. Because of the quantitative aspects of the components (like energy consumption), we need some solution concept to be able to synthesis a component that respect some formal specification and is, in some sense, optimal. In the context of non-zero sum games on graphs for verification and synthesis, Nash equilibria or particular classes of Nash equilibria have been studied for quantitative objectives [6, 3, 4] or qualitative objectives [2]. Recently, J.Y. Halpern and R. Pass defined the notion of *iterated regret minimization* (IRM) [7] in the general framework of (single-shot) *strategic games*, where the players choose in parallel their strategy among a finite number of strategies, and their respective payoffs are given by a finite matrix.

They show that for many games of interest, Nash equilibria suggest strategies that are rejected by common sense, while iterated regret is an alternative solution concept that sometimes proposes more intuitive solutions. In this paper we consider games where the matrix is not given explicitly but defined implicitly

	$A_2$	$B_2$
$A_1$	(2, 1)	(3, 4)
$B_1$	(1, 2)	(4, 3)

**Fig. 1.**

by a game graph and we study the algorithmic aspects of IRM on such games. While it is easy to compute the iterated regret on a finite matrix, it is not that simple for game played on trees or graphs, as the number of strategies (and therefore the underlying matrix) is exponential in the size of a tree and even infinite for a graph.

The IRM solution concept assumes that instead of trying to minimize what she has to pay, each player tries to minimize her *regret*. The regret is informally defined as the difference between what a player actually pays and what she could have payed if she knew the strategy chosen by the other player<sup>1</sup>. More formally, if  $c_1(\lambda_1, \lambda_2)$  represents what Player 1 pays<sup>2</sup> when the pair of strategies  $(\lambda_1, \lambda_2)$  is played,  $\text{reg}_1(\lambda_1, \lambda_2) =$

<sup>1</sup> We only consider 2-player games, but our work can be easily extended to  $n$ -player games.

<sup>2</sup> We could have considered rewards instead of penalties, everything is symmetrical.

$c_1(\lambda_1, \lambda_2) - \min_{\lambda'_1} c_1(\lambda'_1, \lambda_2)$ . Consider the strategic game defined by the matrix of Figure 1. In this game, Player 1 has two strategies  $A_1$  and  $B_1$  and Player 2 has two strategies  $A_2$  and  $B_2$ . The two players choose a strategy at the same time and the pairs of strategies define what the two players have to pay. The regret of playing  $A_1$  for Player 1 when Player 2 plays  $A_2$  is equal to 1 because  $c_1(A_1, A_2)$  is 2 while  $c_1(B_1, A_2)$  is 1. Knowing that Player 2 plays  $A_2$ , Player 1 should have played  $B_1$  instead of  $A_1$ .

As Players have to choose strategies before knowing how the adversary will play, we associate a regret with each strategy as follows. The regret of a strategy  $\lambda_1$  of Player 1 is :  $\text{reg}_1(\lambda_1) = \max_{\lambda_2} \text{reg}_1(\lambda_1, \lambda_2)$ . In the example, the regret attached to strategy  $A_1$  is 1, because when Player 2 plays  $A_2$ , Player 1 regret is 1, and when Player 2 plays  $B_2$  her regret is 0. A rational player should minimize her regret. The regret for Player 1 is thus defined as  $\text{reg}_1 = \min_{\lambda_1} \text{reg}_1(\lambda_1)$ , summarizing, we get  $\text{reg}_1 = \min_{\lambda_1} \max_{\lambda_2} (c_1(\lambda_1, \lambda_2) - \min_{\lambda'_1} c_1(\lambda'_1, \lambda_2))$ . A symmetrical definition can be given for Player 2's regret.

Let us come back to the example. The regret attached to strategy  $B_1$  is 1. So the two strategies of Player 1 are equivalent w.r.t. regret minimization. On the other hand, for Player 2, the regret of  $A_2$  is 0, and the regret of  $B_2$  is 3. So, if Player 1 makes the hypothesis that Player 2 tries to minimize her regret, then she must conclude that Player 2 will play  $A_2$ . Knowing that, Player 1 recomputes her regret for each action, and in this case, the regret of action  $A_1$  is 1 while the regret of  $B_1$  is 0. So rational players minimizing their regret should end up playing  $(B_1, A_2)$  in this game.

Reasoning on rationality is formalized by Halpern and Pass by introducing a *delete operator* that erases strictly dominated strategies. This operator takes sets of strategies  $(A_1, A_2)$  for each player and returns  $D(A_1, A_2) = (A'_1, A'_2)$  the strategies that minimize regret. Then  $D(A'_1, A'_2)$  returns the strategies that minimize regret under the hypothesis that adversaries minimize their regret i.e., choose their strategies in  $A'_1$  and  $A'_2$  resp.

In this paper, we consider games where the matrix is not given explicitly but defined implicitly by a game graph. In particular, we use the same definition of iterated regret as Halpern and Pass. More precisely, we consider graphs where vertices are partitioned into vertices that belong to Player 1 and vertices that belong to Player 2. Each edge is annotated by a cost for Player 1 and one for Player 2. Additionally, there are two designated sets of vertices, one that Player 1 tries to reach and the other one that Player 2 tries to reach. The game starts in the initial vertex of the graph and is played for an infinite number of rounds as follows. In each round, the player who owns the vertex on which the pebble is placed moves the pebble to an adjacent vertex using an edge of the graph, and a new round starts. The infinite plays generate an infinite sequence of vertices and the amount that the players have to pay are computed as follows. Player 1 pays  $+\infty$  if the sequence does not reach the target set assigned to her, otherwise she pays the sum of edge costs assigned to her on the prefix up to the first visit to her target set. The amount to pay for Player 2 is defined symmetrically. Strategies in such games are functions from the set of histories of plays (sequences of visited vertices) to edges (choice of moves for the pebble).

**Contributions** <sup>3</sup> We first consider target-weighted arenas, where the payoff function is defined for each state of the objectives. We give a PTIME algorithm to compute the regret by reduction to a min-max game (and in linear time for trees). We then consider edge-weighted arenas. Each edge is labeled by a pair of integers – one for each player –,

<sup>3</sup> A long version with complete proofs is available at [www.ulb.ac.be/di/ssd/filiot/regret-mfcs.pdf](http://www.ulb.ac.be/di/ssd/filiot/regret-mfcs.pdf)

and the payoffs are defined by the sum of the weights along the path until the first visit to an objective. We give a pseudo-PTIME algorithm to compute the regret in an edge-weighted arena, by reduction to a target-weighted arena. We then study the problem of IRM. We provide a *delete operator* that removes strictly dominated strategies. We show how to compute the effect of iterating this operator on tree arenas and on the general class of graphs where the weights are strictly positive. In the first case, we provide a quadratic time algorithm and in the second case, a pseudo-exponential time algorithm.

**Related works** Regret minimization is a popular notion to define a goal for a learning agent (in the *machine learning* theory). One can also argue that selecting a strategy may be viewed as a learning process. IRM is however not a classical concept of machine learning, since this notion is meaningless when there is a single learning agent. Like [7], our work is thus more related to game theory than learning theory.

[10] investigates how to find a strategy that “mimimizes” the regret in extensive games with imperfect information. This kind of game can be considered as games on a probabilistic finite tree arena. We also consider finite tree arena in Section 5.1. However in that paper, the strategy that achieves the minimal value for regret is not computed, but a strategy with a regret less than a bound depending on the range of utilities, the number of actions and the number of rounds. IRM is not considered neither.

There are several notions of equilibria for reasoning on 2-player non-zero-sum (strategic) games, for instance *Nash equilibrium*, *sequential equilibrium*, *perfect equilibrium* - see [8] for an overview. Those equilibria formalize notions of rational behavior by defining optimality criteria for pairs of strategies. As it has been shown by Halpern and Pass for several examples like the *Centipede game* [9] or the *Traveller’s dilemma* [1], IRM proposes solutions that are more intuitive and natural than Nash equilibria.

## 2 Weighted Games and Regret

Given a cartesian product  $A \times B$  of two sets, we denote by  $\text{proj}_i$  the  $i$ -th projection,  $i = 1, 2$ . It is naturally extended to sequence of elements of  $A \times B$  by  $\text{proj}_i(c_1 \dots c_n) = \text{proj}_i(c_1) \dots \text{proj}_i(c_n)$ . For all  $k \in \mathbb{N}$ , we let  $[k] = \{0, \dots, k\}$ .

**Reachability Games** Turn-based two-player games are played on game arenas by two players. A (finite) *game arena* is a tuple  $G = (S = S_1 \uplus S_2, s_0, T)$  where  $S_1, S_2$  are finite disjoint sets of player positions ( $S_1$  for Player 1 and  $S_2$  for Player 2),  $s_0 \in S$  is the initial position, and  $T \subseteq S \times S$  is the transition relation. A *finite play* on  $G$  of length  $n$  is a finite word  $\pi = \pi_0 \pi_1 \dots \pi_n \in S^*$  such that  $\pi_0 = s_0$  and for all  $i = 0, \dots, n-1$ ,  $(\pi_i, \pi_{i+1}) \in T$ . Infinite plays are defined similarly. We denote by  $\mathbf{P}_f(G)$  (resp.  $\mathbf{P}_\infty(G)$ ) the set of finite (resp. infinite) plays on  $G$ , and we let  $\mathbf{P}(G) = \mathbf{P}_f(G) \cup \mathbf{P}_\infty(G)$ . For any node  $s \in S$ , we denote by  $(G, s)$  the arena  $G$  where the initial position is  $s$ .

Let  $i \in \{1, 2\}$ . We let  $-i = 1$  if  $i = 2$  and  $-i = 2$  if  $i = 1$ . A *strategy*  $\lambda_i : \mathbf{P}_f(G) \rightarrow S \cup \{\perp\}$  for Player  $i$  is a mapping that maps any finite play  $\pi$  whose last position – denoted  $\text{last}(\pi)$  – is in  $S_i$  to  $\perp$  if there is no outgoing edge from  $\text{last}(\pi)$ , and to a position  $s$  such that  $(\text{last}(\pi), s) \in T$  otherwise. The set of strategies of Player  $i$  in  $G$  is denoted by  $\Lambda_i(G)$ . Given a strategy  $\lambda_{-i} \in \Lambda_{-i}(G)$ , the *outcome*  $\text{Out}^G(\lambda_i, \lambda_{-i})$  is the unique play  $\pi = \pi_0 \dots \pi_n \dots$  such that (i)  $\pi_0 = s_0$ , (ii) if  $\pi$  is finite, then there is no outgoing edge from  $\text{last}(\pi)$ , and (iii) for all  $0 \leq j \leq |\pi|$  and all  $\kappa = 1, 2$ , if  $\pi_j \in S_\kappa$ , then  $\pi_{j+1} = \lambda_\kappa(\pi_0 \dots \pi_j)$ . We also define  $\text{Out}^G(\lambda_i) = \{\text{Out}^G(\lambda_i, \lambda_{-i}) \mid \lambda_{-i} \in \Lambda_{-i}(G)\}$ .

A strategy  $\lambda_i$  is *memoryless* if for any play  $h \in \mathbf{P}_f(G)$ ,  $\lambda_i(h)$  only depends on  $\text{last}(h)$ . Thus  $\lambda_i$  can be seen as a function  $S_i \mapsto S \cup \{\perp\}$ . It is *finite-memory* if  $\lambda_i(h)$

1.  $\mathbf{c}_i^G(\pi) = \begin{cases} +\infty & \text{if } \pi \text{ is not winning for Player } i \\ \sum_{j=0}^{\min\{k \mid \pi_k \in \mathbf{C}_i\}-1} \mu_i(\pi_j, \pi_{j+1}) & \text{otherwise} \end{cases}$  *cost of a play*  $\pi$
2.  $\mathbf{c}_i^G(\lambda_1, \lambda_2) = \mathbf{c}_i^G(\text{Out}^G(\lambda_1, \lambda_2))$  *cost of two strategies*  $\lambda_1, \lambda_2$ .
3.  $\text{br}_i^G(\lambda_{-i}) = \min_{\lambda_i \in A_i(G)} \mathbf{c}_i^G(\lambda_i, \lambda_{-i})$  *best response of player*  $i$  *against*  $\lambda_{-i}$
4.  $\text{reg}_i^G(\lambda_i, \lambda_{-i}) = \mathbf{c}_i^G(\lambda_i, \lambda_{-i}) - \text{br}_i^G(\lambda_{-i})$ . *regret of two strategies*  $\lambda_1, \lambda_2$
5.  $\text{reg}_i^G(\lambda_i) = \max_{\lambda_{-i} \in A_{-i}(G)} \text{reg}_i^G(\lambda_i, \lambda_{-i})$  *regret of a strategy*  $\lambda_i$
6.  $\text{reg}_i^G = \min_{\lambda_i \in A_i(G)} \text{reg}_i^G(\lambda_i)$ . *regret of Player*  $i$

**Fig. 2.** Cost, best response, and regret of Player  $i$ ,  $i \in \{1, 2\}$  on a game graph  $G$ , for  $\pi$  a play,  $\lambda_1, \lambda_2$  strategies of Player 1 and 2 respectively.

only depends on last( $h$ ) and on some state of a finite state set. We refer the reader to [5] for formal definitions.

A *reachability winning condition* for Player  $i$  is given by a subset of positions  $\mathbf{C}_i \subseteq S$  – called the *target set* –. A play  $\pi \in \mathbf{P}(G)$  is winning for Player  $i$  if some position of  $\pi$  is in  $\mathbf{C}_i$ . A strategy  $\lambda_i$  for Player  $i$  is *winning* if all the plays of  $\text{Out}^G(\lambda_i)$  are winning. In this paper, we often consider two target sets  $\mathbf{C}_1, \mathbf{C}_2$  for Player 1 and 2 respectively. We write  $(S_1, S_2, s_0, T, \mathbf{C}_1, \mathbf{C}_2)$  to denote the game arena  $G$  extended with those target sets. Finally, let  $\lambda_i \in A_i(G)$  be a winning strategy for Player  $i$  and  $\lambda_{-i} \in A_{-i}(G)$ . Let  $\pi_0 \pi_1 \dots \in \mathbf{P}(G)$  be the outcome of  $(\lambda_i, \lambda_{-i})$ . The outcome of  $(\lambda_i, \lambda_{-i})$  up to  $\mathbf{C}_i$  is defined by  $\text{Out}^{G, \mathbf{C}_i}(\lambda_i, \lambda_{-i}) = \pi_0 \dots \pi_n$  such that  $n = \min\{j \mid \pi_j \in \mathbf{C}_i\}$ . We extend this notation to sets of plays  $\text{Out}^{G, \mathbf{C}_i}(\lambda_i)$  naturally.

**Weighted Games** We add weights on edges of arenas and include the target sets. A (finite) *weighted game arena* is a tuple  $G = (S = S_1 \uplus S_2, s_0, T, \mu_1, \mu_2, \mathbf{C}_1, \mathbf{C}_2)$  where  $(S, s_0, T)$  is a game arena, for all  $i = 1, 2$ ,  $\mu_i : T \rightarrow \mathbb{N}$  is a weight function for Player  $i$  and  $\mathbf{C}_i$  its target set. We let  $M_i^G$  be the maximal weight of Player  $i$ , i.e.  $M_i^G = \max_{e \in T} \mu_i(e)$  and  $M^G = \max(M_1^G, M_2^G)$ .

$G$  is a *target-weighted arena* (TWA for short) if only the edges leading to a target node are weighted by strictly positive integers, and any two edges leading to the same node carry the same weight. Formally, for all  $(s, s') \in T$ , if  $s' \notin \mathbf{C}_i$ , then  $\mu_i(s, s') = 0$ , otherwise for all  $(s'', s') \in T$ ,  $\mu_i(s, s') = \mu_i(s'', s')$ . Thus for target-weighted arenas, we assume in the sequel that the weight functions map  $\mathbf{C}_i$  to  $\mathbb{N}$ .

**Plays, cost, best response and regret** Let  $\pi = \pi_0 \pi_1 \dots \pi_n$  be a finite play in  $G$ . We extend the weight functions to finite plays, so that for all  $i = 1, 2$ ,  $\mu_i(\pi) = \sum_{j=0}^{n-1} \mu_i(\pi_j, \pi_{j+1})$ . Let  $i \in \{1, 2\}$ , the *cost*  $\mathbf{c}_i^G(\pi)$  of  $\pi$  (for Player  $i$ ) is  $+\infty$  if  $\pi$  is not winning for Player  $i$ , and the sum of the weights occurring along the edges defined by  $\pi$  until the first visit to a target position otherwise (see Fig. 2.1). In Fig. 2.2, we extend this notion to the cost of two strategies  $\lambda_1, \lambda_2$  of Player 1 and 2 respectively. The *best response* of Player  $-i$  to  $\lambda_i$ , denoted by  $\text{br}_{-i}^G(\lambda_i)$ , is the least cost Player  $-i$  can achieve against  $\lambda_i$  (Fig. 2.3). The *regret* for Player  $i$  of playing  $\lambda_i$  against  $\lambda_{-i}$  is the difference between the cost Player  $i$  achieves and the best response to the strategy of Player  $-i$  (Fig. 2.4). Note that  $\text{reg}_i^G(\lambda_i, \lambda_{-i}) \geq 0$ , since  $\text{br}_{-i}^G(\lambda_{-i}) \leq \mathbf{c}_i^G(\lambda_i, \lambda_{-i})$ . The

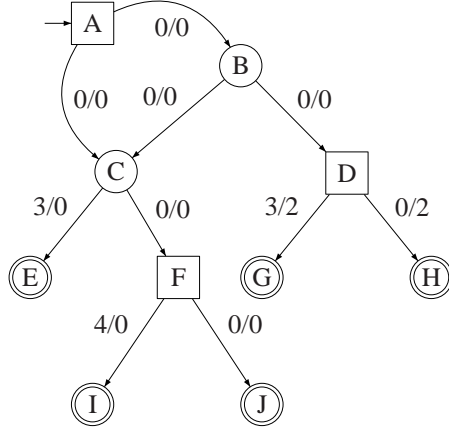
regret of  $\lambda_i$  for Player  $i$  is the maximal regret she gets for all strategies of Player  $-i$  (Fig. 2.5). Finally, the regret of Player  $i$  in  $G$  is the minimal regret she can achieve (Fig. 2.6).

We let  $+\infty - (+\infty) = +\infty$ .

**Proposition 1.**  $\text{reg}_i^G < +\infty$  iff Player  $i$  has a winning strategy,  $i = 1, 2$ .

*Proof.* If Player  $i$  has no winning strategy, then for all  $\lambda_i \in \Lambda_i(G)$ , there is  $\lambda_{-i} \in \Lambda_{-i}(G)$  s.t.  $c_i^G(\lambda_i, \lambda_{-i}) = +\infty$ . Thus  $\text{reg}_i^G(\lambda_i, \lambda_{-i}) = +\infty$ . Therefore  $\text{reg}_i^G = +\infty$ . If Player  $i$  has a winning strategy  $\lambda_i$ , then for all  $\lambda_{-i} \in \Lambda_{-i}(G)$ ,  $c_i^G(\lambda_i, \lambda_{-i}) < +\infty$  and  $\text{br}_i^G(\lambda_{-i}) \leq c_i^G(\lambda_i, \lambda_{-i}) < +\infty$ . Thus  $\text{reg}_i^G \leq \text{reg}_i^G(\lambda_i) < +\infty$ .  $\square$

*Example 1.* Consider the game arena  $G$  of Fig. 3. Player 1's positions are circle nodes and Player 2's positions are square nodes. The target nodes are represented by double circles. The initial node is  $A$ . The weights are given by pairs of integers



**Fig. 3.** Weighted Graph Arena

for Player 1 and 2 respectively. In this example, we are first concerned by computing Player 1's regret.

Let  $\lambda_1$  be the memoryless strategy defined by  $\lambda_1(B) = C$  and  $\lambda_1(C) = E$ . For all  $\lambda_2 \in \Lambda_2(G)$ ,  $\text{Out}_1^G(\lambda_1, \lambda_2)$  is either  $ACE$  or  $ABCE$ , depending on whether Player 2 goes directly to  $C$  or passes by  $B$ . In both cases, the outcome is winning and  $c_1^G(\lambda_1, \lambda_2) = 3$ . What is the regret of playing  $\lambda_1$  for Player 1? To compute  $\text{reg}_1^G(\lambda_1)$ , we should consider all possible strategies of Player 2, but a simple observation allows us to restrict this range. Indeed, to maximize the regret of Player 1, Player 2 should cooperate in subtrees where  $\lambda_1$  prevents to go, i.e. in the subtrees rooted at  $D$  and  $F$ . Therefore we only have to consider the two following memoryless strategies  $\lambda_2$  and  $\lambda_2'$ : both  $\lambda_2$  and  $\lambda_2'$  move from  $F$  to  $J$  and from  $D$  to  $H$ , but  $\lambda_2(A) = B$  while  $\lambda_2'(A) = C$ . In both cases, going to  $F$  is a best response to  $\lambda_2$  and  $\lambda_2'$  for Player 1, i.e.  $\text{br}_1^G(\lambda_2) = \text{br}_1^G(\lambda_2') = 0$ . Therefore we get  $\text{reg}_1^G(\lambda_1, \lambda_2) = c_1^G(\lambda_1, \lambda_2) - \text{br}_1^G(\lambda_2) = 3 - 0 = 3$ . Similarly  $\text{reg}_1^G(\lambda_1, \lambda_2') = 3$ . Therefore  $\text{reg}_1^G(\lambda_1) = 3$ .

As a matter of fact, the strategy  $\lambda_1$  minimizes the regret of Player 1. Indeed, if she chooses to go from  $B$  to  $D$ , then Player 2 moves from  $A$  to  $B$  and from  $D$  to  $G$  (so that Player 1 gets a cost 3) and cooperates in the subtree rooted at  $C$  by moving from  $F$  to  $J$ . The regret of Player 1 is therefore 3. If Player 1 moves from  $B$  to  $C$  and from  $C$  to  $F$ , then Player 2 moves from  $A$  to  $C$  and from  $F$  to  $I$  (so that Player 1 gets a cost 4), and from  $D$  to  $H$ , the regret of Player 1 being therefore 4. Similarly, one can show that all other strategies of Player 1 have a regret at least 3. Therefore  $\text{reg}_1^G = 3$ .

Note that the strategy  $\lambda_1$  does not minimize the regret in the subgame defined by the subtree rooted at  $C$ . Indeed, in this subtree, Player 1 has to move from  $C$  to  $F$ , and the regret of doing this is  $4 - 3 = 1$ . However the regret of  $\lambda_1$  in the subtree is 3. This example illustrates a situation where a strategy that minimizes the regret in the whole game does not necessarily minimize the regret in the subgames. Therefore we cannot apply a simple backward algorithm to compute the regret. As we will see in the next section, we first have to propagate some information in the subgames.

### 3 Regret Minimization on Target-Weighted Graphs

In this section, our aim is to give an algorithm to compute the regret for Player  $i$ . This is done by reduction to a min-max game, defined in the sequel. We say that we *solve* the regret minimization problem (RMP for short) if we can compute the minimal regret and a (finite representation of a) strategy that achieves this value. We first introduce the notion of games with minmax objectives. Let  $G = (S = S_1 \uplus S_2, s_0, T, \mu_1, \mu_2, \mathbf{C}_1, \mathbf{C}_2)$  be a TWA and  $i = 1, 2$ . We let  $\text{minmax}_i^G = \min_{\lambda_i \in A_i(G)} \max_{\lambda_{-i} \in A_{-i}(G)} \mathbf{c}_i^G(\lambda_i, \lambda_{-i})$ .

**Proposition 2.** *Given a TWA  $G = (S, s_0, T, \mu_1, \mu_2, \mathbf{C}_1, \mathbf{C}_2)$ ,  $i \in \{1, 2\}$  and  $K \in \mathbb{N}$ , one can decide in time  $O(|S|+|T|)$  whether  $\text{minmax}_i^G \leq K$ .  $\text{minmax}_i^G$  and a memory-less strategy that achieves this value can be computed in time  $O(\log_2(M_i^G)(|S|+|T|))$ .*

Since roles of the players are symmetric, without loss of generality we can focus on computing the regret of Player 1 only. Therefore we do not consider Player 2's targets and weights. Let  $G = (S = S_1 \uplus S_2, s_0, T, \mu_1, \mathbf{C}_1)$  be a TWA (assumed to be fixed from now on). Let  $\lambda_1 \in A_1(G)$  be a winning strategy of Player 1 (if it exists). Player 2 can enforce Player 1 to follow one of the paths of  $\text{Out}^{G, \mathbf{C}_1}(\lambda_1)$  by choosing a suitable strategy. When choosing a path  $\pi \in \text{Out}^{G, \mathbf{C}_1}(\lambda_1)$ , in order to maximize the regret of Player 1, Player 2 cooperates (i.e. she minimizes the cost) if Player 1 would have deviated from  $\pi$ . This leads to the notion of *best alternative* along a path. Informally, the best alternative along  $\pi$  is the minimal value Player 1 could have achieved if she deviated from  $\pi$ , assuming Player 2 cooperates. Since Player 2 can enforce one of the paths of  $\text{Out}^{G, \mathbf{C}_1}(\lambda_1)$ , to maximize the regret of Player 1, she will choose the path  $\pi$  with the highest difference between  $\mathbf{c}_1^G(\pi)$  and the minimal best alternative along  $\pi$ . As an example consider the TWA arena of Fig. 3. In this example, if Player 1 moves from  $C$  to  $E$ , then along the path  $ACE$ , the best alternative is 0. Indeed, the other alternative was to go from  $C$  to  $F$  and in this case, Player 2 would have cooperated.

We now formally define the notion of best alternative. Let  $s \in S_1$ . The best value that can be achieved from  $s$  by Player 1 when Player 2 cooperates is defined by:

$$\text{best}_1^G(s) = \min_{\lambda_1 \in A_1(G, s)} \min_{\lambda_2 \in A_2(G, s)} \mathbf{c}_1^{(G, s)}(\lambda_1, \lambda_2)$$

Let  $(s, s') \in T$ . The *best alternative* of choosing  $s'$  from  $s$  for Player 1, denoted by  $\text{ba}_1^G(s, s')$ , is defined as the minimal value she could have achieved by choosing another successor of  $s$  (assuming Player 2 cooperates). Formally:

$$\text{ba}_1^G(s, s') = \begin{cases} +\infty & \text{if } s \in S_2 \\ \min_{(s, s'') \in T, s'' \neq s'} \text{best}_1^G(s'') & \text{if } s \in S_1 \end{cases}$$

with  $\min \emptyset = +\infty$ . Finally, the best alternative of a path  $\pi = s_0 s_1 \dots s_n$  is defined as  $+\infty$  if  $n = 0$  and as the minimal best alternative of the edges of  $\pi$  otherwise:

$$\text{ba}_G^1(\pi) = \min_{0 \leq j < n} \text{ba}_G^1(s_j, s_{j+1})$$

We first transform the graph  $G$  into a graph  $G'$  such that all the paths that lead to a node  $s$  have the same best alternative. This can be done since the number of best alternatives is bounded by  $|\mathbf{C}_1|$ . The construction of  $G'$  is done inductively by storing the best alternatives in the positions.

**Definition 1.** The graph of best alternatives of  $G$  is the TWA  $G' = (S' = S'_1 \uplus S'_2, s'_0, T', \mu'_1, \mathbf{C}'_1)$  where  $S'_i = S_i \times ([M_1^G] \cup \{+\infty\})$ ,  $s'_0 = (s_0, +\infty)$ ,  $\mathbf{C}'_1 = S'_1 \cap (C_1 \times [M_1^G])$ , for all  $(s, b) \in \mathbf{C}'_1$ ,  $\mu'_1(s, b) = \mu_1(s)$ , and for all  $(s, b_1), (s', b'_1) \in S'$ ,  $((s, b_1), (s', b'_1)) \in T'$  iff  $(s, s') \in T$  and

$$b'_1 = \begin{cases} \min(b_1, \mathbf{ba}_1^G(s, s')) & \text{if } s \in S_1 \\ b_1 & \text{if } s \in S_2 \end{cases}$$

The best alternative along the paths that lead to the same node is unique. Moreover, as the number of best alternatives is bounded by  $|\mathbf{C}_1|$ ,  $G'$  can be constructed in PTime:

**Proposition 3.** For all  $(s, b) \in S'$  and all finite path  $\pi$  in  $G'$  from  $(s_0, +\infty)$  to  $(s, b)$ ,  $\mathbf{ba}_1^{G'}(\pi) = b$ .  $G'$  can be constructed in time  $O((|\mathbf{C}_1| + \log_2(M_1^G)) \times (|S| + |T|))$ .

Since the best alternative information depends only on the paths, the paths of  $G$  and those of  $G'$  are in bijection. This bijection can be extended to strategies. In particular, we define two mappings  $\Phi_i$  from  $\Lambda_i(G)$  to  $\Lambda_i(G')$ , for all  $i = 1, 2$ . For all path  $\pi = s_0 s_1 \dots$  in  $G$  (finite or infinite), we denote by  $B(\pi)$  the path of  $G'$  defined by  $(s_0, b_0)(s_1, b_1) \dots$  where  $b_0 = +\infty$  and for all  $j > 0$ ,  $b_j = \mathbf{ba}_1^G(s_0 \dots s_{j-1})$ . The mapping  $B$  is bijective, and its inverse corresponds to  $\mathbf{proj}_1$ .

The mapping  $\Phi_i$  maps any strategy  $\lambda_i \in \Lambda_i(G)$  to a strategy  $\Phi_i(\lambda_i) \in \Lambda_i(G')$  such that  $\Phi_i(\lambda_i)$  behaves as  $\lambda_i$  on the first projection of the play and adds the best alternative information to the position. Let  $h \in S'^*$  such that  $\text{last}(h) \in S'_i$ . Let  $s = \lambda_i(\mathbf{proj}_1(h))$ . Then  $\Phi_i(\lambda_i)(h) = (s, \mathbf{ba}_1^G(\mathbf{proj}_1(h), s))$ . The inverse mapping  $\Phi_i^{-1}$  just projects the best alternative information. In particular, for all  $\lambda'_i \in \Lambda_i(G')$ , and all  $h \in S'^*$  such that  $\text{last}(h) \in S_i$ ,  $\Phi_i^{-1}(\lambda'_i)(h) = \mathbf{proj}_1(\lambda'_i(B(h)))$ . Clearly, the mappings  $\Phi_i$  are bijective.

The best alternative information is crucial. This is a global information that allows us to compute the regret locally. For all  $(s, b) \in \mathbf{C}'_1$ , we let  $\nu_1(s, b) = \mu_1(s) - \min(\mu_1(s), b)$ , and extend  $\nu_1$  to strategies naturally ( $\nu_1(\lambda_1, \lambda_2) = +\infty$  if  $\lambda_1$  is losing).

**Lemma 1.** Let  $H = (S', s'_0, T', \nu^1, \mathbf{C}'_1)$  where  $S', s'_0, T', \mathbf{C}'_1$  are defined in Definition 1. For all  $\lambda_1 \in \Lambda_1(G)$  and all  $\lambda'_1 \in \Lambda_1(G')$ , the following holds:

$$1. \text{reg}_1^G(\lambda_1) = \text{reg}_1^{G'}(\Phi_1(\lambda_1)) \quad 2. \text{reg}_1^{G'}(\lambda'_1) = \max_{\lambda'_2 \in \Lambda_2(G')} \nu_1(\lambda'_1, \lambda'_2) \quad 3. \text{reg}_1^G = \min \max_1^H$$

*Proof.* 1 and 2 are in the full version of the paper. For 3, we have

$$\begin{aligned} \text{reg}_1^G &= \min_{\lambda_1 \in \Lambda_1(G)} \text{reg}_1^G(\lambda_1) \text{ (by definition)} = \min_{\lambda_1 \in \Lambda_1(G)} \text{reg}_1^H(\Phi_1(\lambda_1)) && \text{(by 1)} \\ &= \min_{\lambda_1 \in \Lambda_1(H)} \text{reg}_1^H(\lambda_1) \text{ (by 1)} && = \min_{\lambda_1 \in \Lambda_1(H)} \max_{\lambda_2 \in \Lambda_2(H)} \nu^1(\lambda_1, \lambda_2) \text{ (by 2)} \square \end{aligned}$$

As a consequence of Lemma 1, we can solve the RMP on TWA's. We first compute the graph of best alternatives and solve a minmax game. This gives us a memoryless strategy that achieves the minimal regret in the graph of best alternatives. To compute a strategy in the original graph, we apply the inverse mapping  $\Phi_1^{-1}$ : this gives a finite-memory strategy whose memory is exactly the best alternative seen along the current finite play. Therefore the needed memory is bounded by the number of best alternatives, which is bounded by  $|\mathbf{C}_1|$ .

**Theorem 1.** The RMP on a TWA  $G = (S, s_0, T, \mu_1, \mathbf{C}_1)$  can be solved in time  $O(|\mathbf{C}_1| \cdot \log_2(M_1^G) \cdot (|S| + |T|))$ , where  $M_1^G = \max_{e \in T} \mu_i(e)$ .

## 4 Regret Minimization in Edge-Weighted Graphs

In this section, we give a pseudo-polynomial time algorithm to solve the RMP in weighted arenas (with weights on edges). In a first step, we prove that if the regret is finite, the strategies minimizing the regret generates outcomes whose cost is bounded by some value which depends on the graph. This allows us to reduce the problem to the RMP in a TWA, which can then be solved by the algorithm of the previous section.

Let  $G = (S = S_1 \uplus S_2, s_0, T, \mu_1, \mathbf{C}_1)$  be a weighed game arena with objective  $\mathbf{C}_1$ . As in the previous section, we assume that we want to minimize the regret of Player 1, so we omit the weight function and the target of Player 2.

**Definition 2 (Bounded strategies).** Let  $B \in \mathbb{N}$  and  $\lambda_1 \in \Lambda_1(G)$ . The strategy  $\lambda_1$  is bounded by  $B$  if for all  $\lambda_2 \in \Lambda_2(G)$ ,  $\mathbf{c}_1^G(\lambda_1, \lambda_2) \leq B$ .

Note that a bounded strategy is necessarily winning, since by definition, the cost of some outcome is infinite iff it is loosing. The following lemma states that the winning strategies that minimize the regret of Player 1 are bounded.

**Lemma 2.** For all weighted arena  $G = (S, s_0, T, \mu_1, \mathbf{C}_1)$  and for any strategy  $\lambda_1 \in \Lambda_1(G)$  winning in  $G$  for Player 1 that minimizes her regret,  $\lambda_1$  is bounded by  $2M^G|S|$ .

*Proof (Sketch).* If there is a winning strategy, there is a memoryless winning strategy  $\gamma_1$  ([5]). Then  $\text{reg}_1^G \leq \text{reg}_1^G(\gamma_1)$ . For any  $\lambda_2$ ,  $\text{Out}^{G, \mathbf{C}_1}(\gamma_1, \lambda_2)$  does not contain twice the same position. Thus  $\mathbf{c}_G^1(\gamma_1, \lambda_2) \leq M^G|S|$ . Moreover,  $\text{br}_1^G(\lambda_2) \leq \mathbf{c}_G^1(\gamma_1, \lambda_2) \leq M^G|S|$ . Thus  $\text{reg}_1^G \leq \text{reg}_1^G(\gamma_1, \lambda_2) \leq M^G|S|$ . Let  $\lambda_1$  minimizing  $\text{reg}_1^G(\lambda_1)$ . Then  $\text{reg}_1^G(\lambda_1, \lambda_2) \leq M^G|S|$ , so  $\mathbf{c}_1^G(\lambda_1, \lambda_2) \leq M^G|S| + \text{br}_1(\lambda_2) \leq 2M^G|S|$ , for any  $\lambda_2$ .

Let  $B = 2M^G|S|$ . Thanks to Lemma 2 we can reduce the RMP in a weighted arena into the RMP in a TWA. Indeed, it suffices to enrich every position of the arena with the sum of the weights occuring along the path used to reach this position. A position may be reachable by several paths, therefore it will be duplicated as many times as they are different path utilities. This may be unbounded, but Lemma 2 ensures that it is sufficient to sum the weights up to  $B$  only. This may results in a larger graph, but its size is still pseudo-polynomial (polynomial in the maximal weight and the size of the graph).

**Definition 3.** Let  $G = (S = S_1 \uplus S_2, s_0, T, \mu_1, \mathbf{C}_1)$  be a weighed game arena. The graph of cost is the TWA  $G' = (S' = S'_1 \uplus S'_2, s'_0, T', \mu'_1, \mathbf{C}'_1)$  defined by: (i)  $S'_i = S_i \times [B]$  and  $s'_0 = (s_0, 0)$ ; (ii) for all  $(s, u), (s', u') \in S'$ ,  $((s, u), (s', u')) \in T'$  iff  $(s, s') \in T$  and  $u' = u + \mu_1(s, s')$ ; (iii)  $\mathbf{C}'_1 = (\mathbf{C}_1 \times [B]) \cap S'$  and  $\forall (s, u) \in \mathbf{C}'_1$ ,  $\mu'_1(s, u) = u$ .

**Lemma 3.**  $\text{reg}_1^G = \text{reg}_1^{G'}$

*Proof (Sketch).* We define a mapping  $\Phi$  that maps the strategies of Player  $i$  in  $G$  to the strategies of Player  $i$  in  $G'$ , for all  $i \in \{1, 2\}$ , which satisfies  $\Phi(\Lambda_i(G)) = \Lambda_i(G')$  and preserves the regret of Player 1's strategies bounded by  $B$ , i.e.  $\text{reg}_1^G(\lambda_1) = \text{reg}_1^{G'}(\Phi(\lambda_1))$ , for all  $\lambda_1 \in \Lambda_1(G)$  bounded by  $B$ .  $\square$

To solve the RMP for a weighted arena  $G$ , we first construct the graph of cost  $G'$ , and then apply Theorem 1, since  $G'$  is a TWA. Correctness is ensured by Lemma 3. This returns a finite-memory strategy of  $G'$  that minimizes the regret, whose memory is the best alternative seen so far. To obtain a strategy of  $G$  minimizing the regret, one applies the inverse mapping  $\Phi^{-1}$ . This gives us a finite-memory strategy whose memory is the cost of the current play up to  $M^G$  and the best alternative seen so far.

**Theorem 2.** *The RMP on a weighted arena  $G = (S = S_1 \uplus S_2, s_0, T, \mu_1, \mathbf{C}_1)$  can be solved in time  $O((M^G)^2 \cdot \log_2(|S| \cdot M^G) \cdot |S| \cdot \mathbf{C}_1 \cdot (|S| + |T|))$ .*

## 5 Iterated Regret Minimization (IRM)

In this section, we show how to compute the iterated regret for tree arenas and for weighted arenas where weights are strictly positive (by reduction to a tree arena).

Let  $G = (S = S_1 \uplus S_2, s_0, T, \mu_1, \mu_2, \mathbf{C}_1, \mathbf{C}_2)$  be a weighted arena. Let  $i \in \{1, 2\}$ ,  $P_i \subseteq \Lambda_i(G)$  and  $P_{-i} \subseteq \Lambda_{-i}(G)$ . The regret of Player  $i$  when she plays strategies of  $P_i$  and when Player  $-i$  plays strategies of  $P_{-i}$  is defined by:

$$\begin{aligned} \text{reg}_i^{G, P_i, P_{-i}} &= \min_{\lambda_i \in P_i} \max_{\lambda_{-i} \in P_{-i}} \mathbf{c}_i^G(\lambda_i, \lambda_{-i}) - \text{br}_i^{G, P_i}(\lambda_{-i}) \\ \text{br}_i^{G, P_i}(\lambda_{-i}) &= \min_{\lambda_i^* \in P_i} \mathbf{c}_i^G(\lambda_i^*, \lambda_{-i}) \end{aligned}$$

For all  $\lambda_i \in P_i$  and  $\lambda_{-i} \in P_{-i}$ , we define  $\text{reg}_i^{G, P_i, P_{-i}}(\lambda_i)$  and  $\text{reg}_i^{G, P_i, P_{-i}}(\lambda_i, \lambda_{-i})$  accordingly. We now define the strategies of rank  $j$ , which are the ones that survived  $j$  times the deletion of strictly dominated strategies. The strategies of rank 0 for Player  $i$  is  $\Lambda_i(G)$ . The strategies of rank 1 for both players are those which minimize their regret against strategy of rank 0. More generally, the strategies of rank  $j$  for Player  $i$  are the strategies of rank  $j-1$  which minimize her regret against Player  $-i$ 's strategies of rank  $j-1$ . Formally, let  $i \in \{1, 2\}$ ,  $P_1 \subseteq \Lambda_1(G)$  and all  $P_2 \subseteq \Lambda_2(G)$ . Then  $D_i(P_1, P_2)$  is the set of strategies  $\lambda_i \in P_i$  such that  $\text{reg}_i^{G, P_i, P_{-i}} = \text{reg}_i^{G, P_i, P_{-i}}(\lambda_i)$ . Then the strategies of rank  $j$  are obtained via a *delete* operator  $D : 2^{\Lambda_1(G)} \times 2^{\Lambda_2(G)} \rightarrow 2^{\Lambda_1(G)} \times 2^{\Lambda_2(G)}$  such that  $D(P_1, P_2) = (D_1(P_1, P_2), D_2(P_1, P_2))$ . We let  $D^j = \underbrace{D \circ \dots \circ D}_{j \text{ times}}$ .

**Definition 4 ( $j$ -th regret).** *Let  $j \geq 0$ . The set of strategies of rank  $j$  for Player  $i$  is  $P_i^j = \text{proj}_i(D^j(\Lambda_1(G), \Lambda_2(G)))$ . The  $j+1$ -th regret for Player  $i$  is defined by  $\text{reg}_i^{G, j+1} = \text{reg}_i^{G, P_i^j, P_{-i}^j}$ . In particular,  $\text{reg}_i^{G, 1} = \text{reg}_i^G$ .*

**Proposition 4.** *Let  $i \in \{1, 2\}$ . For all  $j \geq 0$ ,  $P_i^{j+1} \subseteq P_i^j$  and  $\text{reg}_i^{G, j+1} \leq \text{reg}_i^{G, j}$ . Moreover, there is  $\star \geq 1$  such that for all  $j \geq \star$ , for all  $i \in \{1, 2\}$ ,  $\text{reg}_i^{G, j} = \text{reg}_i^{G, \star}$ .*

**Definition 5 (iterated regret).** *For all  $i = 1, 2$ , the iterated regret of Player  $i$  is  $\text{reg}_i^{G, \star}$ .*

*Example 2.* Consider example Fig. 3. We already saw that the strategies that minimize Player 1's regret are  $B \mapsto C \mapsto E$  and  $B \mapsto D$ , in which cases we have  $\text{reg}_1^G = \text{reg}_1^{G, 1} = 3$ . The strategies that minimize Player 2's regret are  $\lambda_2 : A \mapsto C, F \mapsto I$  and  $\lambda_2' : A \mapsto C, F \mapsto J$ , in which cases her regret is 0. If Player 1 knows that Player 2 plays according to  $\lambda_2$  or  $\lambda_2'$ , she can still play  $C \mapsto E$  but now her regret is 0, so that  $\text{reg}_1^{G, \star} = 0$ . Similarly,  $\text{reg}_2^{G, \star} = 0$ .

### 5.1 IRM in Tree Arenas

In this section, we let  $i \in \{1, 2\}$  and  $G = (S = S_1 \uplus S_2, s_0, T, \mu_1, \mu_2, \mathbf{C}_1, \mathbf{C}_2)$  be a finite edge-weighted tree arena. We can transform  $G$  into a target-weighted tree arena such that  $\mathbf{C}_1 = \mathbf{C}_2$  (denoted by  $\mathbf{C}$  in the sequel) is the set of leaves of the tree, if we allow the functions  $\mu_i$  to take the value  $+\infty$ . This transformation results in a new target-weighted tree arena  $G' = (S = S_1 \uplus S_2, s_0, T, \mu'_1, \mu'_2, \mathbf{C})$  with the same set of states and transitions as  $G$  and for all leaf  $s \in \mathbf{C}$ ,  $\mu'_i(s) = \mathbf{c}_i^{G'}(\pi)$ , where  $\pi$  is the root-to-leaf path leading to  $s$ . The time complexity of this transformation is  $O(|S|)$ .

We now assume that  $G = (S = S_1 \uplus S_2, s_0, T, \mu_1, \mu_2, \mathbf{C})$  is a target-weighted tree arena where  $\mathbf{C}$  is the set of leaves. Our goal is to define a delete operator  $D$  such that  $D(G)$  is a subtree of  $G$  such that for all  $i = 1, 2$ ,  $\Lambda_i(D(G))$  are the strategies of  $\Lambda_i(G)$  that minimize  $\text{reg}_i^G$ . In other words, any pairs of subsets of strategies for both players in  $G$  can be represented by a subtree of  $G$ . This is possible since all the strategies in a tree arena are memoryless. A set of strategies  $P_i \subseteq \Lambda_i(G)$  is therefore represented by removing from  $G$  all the edges  $(s, s')$  such that there is no strategy  $\lambda_i \in P_i$  such that  $\lambda_i(s) = s'$ . In our case, one first computes the set of strategies that minimize regret. This is done as in Section 3 by constructing the tree of best alternatives  $H$  (but in this case with the best alternative of both players) and by solving a min-max game. From  $H$  we delete all edges that are not compatible with a strategy that minimize the minmax value of some player. We obtain therefore a subtree  $D(H)$  of  $H$  such that any strategy of  $H$  is a strategy of  $D(H)$  for Player  $i$  iff it minimizes the minmax value in  $H$  for Player  $i$ . By projecting away the best alternative information in  $D(H)$ , we obtain a subtree  $D(G)$  of  $G$  such that any Player  $i$ 's strategy of  $G$  is a strategy of  $D(G)$  iff it minimizes Player  $i$ 's regret in  $G$ . We can iterate this process to compute the iterated regret, and we finally obtain a subtree  $D^*(G)$  such that any strategy of  $G$  minimizes the iterated regret for Player  $i$  iff it is a Player  $i$ 's strategy in  $D^*(G)$ .

**Definition 6.** *The tree of best alternatives of  $G$  is the tree  $H = (S'_1, S'_2, s'_0, T', \mu'_1, \mu'_2, \mathbf{C}')$  where: (i)  $S' = S'_1 \uplus S'_2$  with  $S'_i = \{(s, b_1, b_2) \mid s \in S_i, b_\kappa = \mathbf{ba}_\kappa^G(\pi_s), \kappa = 1, 2\}$  where  $\pi_s$  is the path from the root  $s_0$  to  $s$ , (ii)  $s'_0 = (s_0, +\infty, +\infty)$ , (iii) for all  $s, s' \in S'$ ,  $(s, s') \in T'$  iff  $(\text{proj}_1(s), \text{proj}_1(s')) \in T$ , (iv)  $\mathbf{C}' = \{s \in S' \mid \text{proj}_1(s) \in \mathbf{C}\}$ , (v) for all  $(s, b_1, b_2) \in \mathbf{C}'$ ,  $\mu'_i(s, b_1, b_2) = \mu_i(s) - \min(\mu_i(s), b_i)$ .*

Note that  $H$  is isomorphic to  $G$ . There is indeed a one-to-one mapping  $\Phi$  between the states of  $G$  and the states of  $H$ : for all  $s \in S$ ,  $\Phi(s)$  is the only state  $s' \in S'$  of the form  $s' = (s, b_1, b_2)$ . Moreover, this mapping is naturally extended to strategies. Since all strategies are memoryless, any strategy  $\lambda_i \in \Lambda_i(G)$  is a function  $S_i \rightarrow S$ . Thus, for all  $s' \in S'_i$ ,  $\Phi(\lambda_i)(s') = \Phi(\lambda_i(\Phi^{-1}(s')))$ . Without loss of generality and for a technical reason, we assume that any strategy  $\lambda_i$  is only defined for states  $s \in S_i$  that are compatible with this strategy, i.e. if  $s$  is not reachable under  $\lambda_i$  then the value of  $\lambda_i$  does not need to be defined. The lemmas of Section 3 still hold for the tree  $H$ . In particular, for all  $i \in \{1, 2\}$ ,  $\Phi(\Lambda_i(G)) = \Lambda_i(H)$  and any strategy  $\lambda_i \in \Lambda_i(G)$  minimizes  $\text{reg}_i^G$  iff  $\Phi(\lambda_i)$  minimizes  $\text{minmax}_i^H$ . Moreover  $\text{reg}_i^G = \text{minmax}_i^H$ .

As in Section 3, the RMP on a tree arena can be solved by a min-max game. For all  $s \in S'$ , we define  $\text{minmax}_i^H(s) = \text{minmax}_i^{(H,s)}$  and compute these values inductively by a bottom-up algorithm that runs in time  $O(|S|)$ . This algorithm not only allows us to compute  $\text{minmax}_i^H$  for all  $i \in \{1, 2\}$ , but also to compute a subtree  $D(H)$  that represents all Player  $i$ 's strategies that achieve this value. We actually define the

operator  $D$  in two steps. First, we remove the edges  $(s, s') \in T'$ , such that  $s \in S'_i$  and  $\min\max_i^H(s') > \min\max_i^H$  for all  $i = 1, 2$ . We obtain a new graph  $H'$  consisting of several disconnected tree components. In particular, there are some states no longer reachable from the root  $s'_0$ . Then we keep the connected component that contains  $s'_0$  and obtain a new tree  $D(H)$ . Since there is a one-to-one correspondence between the strategies minimizing the regret in  $G$  and the strategies minimizing the minmax value in  $H$ , we can define  $D(G)$  by applying to  $D(H)$  the isomorphism  $\Phi^{-1}$ , in other words by projecting the best alternatives away, and by restoring the functions  $\mu_i$ .

We obtain a new tree  $D(G)$  whose Player  $i$ 's strategies minimize the regret of Player  $i$ , for all  $i = 1, 2$ . We can iterate the regret computation on  $D(G)$  and get the Player  $i$ 's strategies that minimize the regret of rank 2 of Player  $i$ , for all  $i = 1, 2$ . We continue iteration until we get a tree  $G'$  such that  $D(G') = G'$ . We let  $D^0(G) = G$  and  $D^{j+1}(G) = D(D^j(G))$ .

**Proposition 5.** *Let  $i \in \{1, 2\}$  and  $j > 0$ . We have  $\text{reg}_i^{G,j} = \text{reg}_i^{D^{j-1}(G)}$ .*

**Theorem 3.** *Let  $G = (S = S_1 \uplus S_2, s_0, T, \mu_1, \mu_2, \mathbf{C})$  be a tree arena. For all  $i = 1, 2$ , the iterated regret of Player  $i$ ,  $\text{reg}_i^{G,*}$ , can be computed in time  $O(|S|^2)$ .*

*Proof.* There is an integer  $j$  such that  $\text{reg}_i^{G,*} = \text{reg}_i^{D^j(G)}$ . According to the definition of  $D(G)$ ,  $j \leq |S|$  because we remove at least one edge of the tree at each step. Since  $D(G)$  can be constructed in time  $O(|S|)$ , the whole time complexity is  $O(|S|^2)$ .  $\square$

## 5.2 IRM in Positive Weighted Arenas

A weighted arena  $G$  is said to be *positive* if all edges are weighted by strictly positive weights only. In this section, we let  $G = (S = S_1 \uplus S_2, s_0, T, \mu_1, \mu_2, \mathbf{C}_1, \mathbf{C}_2)$  be a positive weighted arena. Remind that  $P_i^j(G)$  is the set of strategies that minimize  $\text{reg}_i^{G,j}$ , for all  $j \geq 0$  and  $i = 1, 2$ . As for the regret computation in edge-weighted graphs, we define a notion of boundedness for strategies. Then the iterated regret is computed on the unfolding of the graph, up to some cost bound.

**Definition 7 ( $j$ -winning and  $j$ -bounded strategies).** *Let  $i \in \{1, 2\}$  and  $\lambda_i \in \Lambda_i(G)$ . The strategy  $\lambda_i$  is  $j$ -winning if for all  $\lambda_{-i} \in P_{-i}^j(G)$ ,  $\text{Out}^G(\lambda_i, \lambda_{-i})$  is winning. It is  $j$ -bounded by some  $B \geq 0$  if it is  $j$ -winning, and for all  $\lambda_{-i} \in P_{-i}^j(G)$  and all  $\kappa \in \{i, -i\}$ ,  $\mu_\kappa(\text{Out}^{G, \mathbf{C}_i}(\lambda_i, \lambda_{-i})) \leq B$ .*

Note that  $j$ -boundedness differs from boundedness as we require that the utilities of both players are bounded. We let  $b^G = 6(M^G)^3|S|$ . We get a similar result than the boundedness of strategies that minimize the regret of rank 1, but for any rank:

**Lemma 4.** *For all  $i = 1, 2$  and all  $j \geq 0$ , all  $j$ -winning strategies of Player  $i$  which minimize the  $(j + 1)$ -th regret are  $j$ -bounded by  $b^G$ .*

Lemma 4 allows us to reduce the problem to the IRM in a weighted tree arena, by unfolding the graph arena  $G$  up to some maximal cost value. Lemma 4 suggests to take  $b^G$  for this maximal value. However the best responses to a strategy  $j$ -bounded by  $b^G$  are not necessarily bounded by  $b^G$ , but they are necessarily  $j$ -bounded by  $b^G \cdot M^G$ , since the weights are strictly positive. Therefore we let  $B^G = b^G \cdot M^G$  and take  $B^G$

as the maximal value. Since the  $j$ -winning strategies are  $j$ -bounded by  $b^G$  and the best responses are  $j$ -bounded by  $B^G$ , we do not lose information by taking the unfolding up to  $B^G$ . Finally we apply Theorem 3 on the unfolding. One of the most technical result is to prove the correctness of this reduction.

**Theorem 4.** *The iterated regret for both players in a positive weighted arena  $G$  can be computed in pseudo-exponential time (exponential in  $|S|$ ,  $|T|$  and  $M^G$ ).*

For all  $i = 1, 2$ , the procedure of Section 5.1 returns a finite-memory strategy  $\lambda_i$  minimizing the iterated regret in  $G'$  whose memory is the best alternatives seen so far by both players. From  $\lambda_i$  we can compute a finite-memory strategy in  $G$  minimizing the iterated regret of Player  $i$ , the needed memory is the best alternatives seen by both players and the current finite play up to  $B^G$ . When the cost is greater than  $B^G$ , then any move is allowed. Therefore one needs to add one more bit of memory expressing whether the cost is greater than  $B^G$ .

**Conclusion** The theory of infinite qualitative non-zero sum games over graphs is still in an initial development stage. We adapted a new solution concept from strategic games to game graphs, and gave algorithms to compute the regret and iterated regret. The strategies returned by those algorithms have a finite memory. One open question is to know whether this memory is necessary. In other words, are memoryless strategies sufficient to minimize the (iterated) regret in game graphs? Another question is to determine a lower bound on the complexity of (iterated) regret minimization. Iterated regret minimization over the full class of graphs is still open. In the case of (strictly) positive arenas, the unfolding of the graph arena up to some cost bound can be seen as a finite representation of (possibly infinite) set of strategies of rank  $j$  in  $G$ . Finding such a representation is not obvious for the full class of weighted arenas, since before reaching its objective, a player can take a 0-cost loop finitely many times without affecting her minimal regret. This suggests to add *fairness* conditions on edges to compute the IR.

## References

1. K. Basu. The traveler's dilemma: Paradoxes of rationality in game theory. *American Economic Review*, 84(2):391–95, 1994.
2. T. Brihaye, V. Bruyère, and J. De Pril. Equilibria in quantitative reachability games. In *International Computer Science Symposium in Russia*, 2010.
3. K. Chatterjee, T. A. Henzinger, and M. Jurdzinski. Games with secure equilibria. In *LICS*, pages 160–169, 2004.
4. D. Fisman, O. Kupferman, and Y. Lustig. Rational synthesis. In *TACAS*, pages 190–204, 2010.
5. E. Grädel, W. Thomas, and T. Wilke, editors. *Automata, Logics and Infinite Games*, volume 2500 of *Lecture Notes in Computer Science*. Springer, 2002.
6. E. Grädel and M. Ummels. Solution concepts and algorithms for infinite multiplayer games. In *New Perspectives on Games and Interaction*, volume 4 of *Texts In Logic and Games*, pages 151–178, 2008.
7. J. Y. Halpern and R. Pass. Iterated regret minimization: A more realistic solution concept. In *IJCAI*, 2009.
8. M.J. Osborne and A. Rubinstein. *A Course in Game Theory*. MIT Press, 1994.
9. R. W. Rosenthal. Games of perfect information, predatory pricing and the chain-store paradox. *Journal of Economic Theory*, 25(1):92–100, 1981.
10. M. Zinkevich, M. Johanson, M. Bowling, and C. Piccione. Regret minimization in games with incomplete information. In *NIPS*, 2007.