

Regret Minimization of Extensive Games and Its Application on Game Strategies

Jie Ren *

Department of Mathematics and applied mathematics, Zhejiang University, Zhejiang 310058, China

* Corresponding Author Email: 3190105701@zju.edu.cn

Abstract. Game theory has been discussed by people from generation to generation. The concept of game theory has been applied to various areas. Extensive games, as a typical form of games, are able to simulate the circumstances in many other areas. Recently, more and more attention has been paid to finding a Nash equilibrium in large extensive games. In this paper, we describe a method to solve extensive games, the basis of this method is regret minimization. We then use a concrete case to demonstrate how this technique can be used to make decisions by players in the game of Hearthstone. We calculate the overall regret of all the strategies that the player can take in a given circumstance, and then choose the strategy with the least overall regret. We demonstrate through the case that by minimizing regret in extensive game models, players are able to optimize their strategies and increase the chances to win.

Keywords: game theory; extensive games; regret minimization

1. Introduction

Game theory describes how a player chooses actions in order to get the best possible result for himself, while expecting reasonable actions from other players. A player is the basic part of a game that decides and acts. A game is a precise description of the strategic interaction that includes the constraints of, and payoffs for, actions that the players can take, but says nothing about what actions they actually take [1].

The concept of game theory has found application in various areas like economics, finance, sciences, politics, auction and cyber security. Hamilton et al. [2], outline how game theory can be used in information warfare. Chakrabarti focuses on the security (attack and defense) on the Internet and its infrastructure [3]. A game-theoretic framework is proposed by Liu et al. [4] to analyze the interactions between attacking and defending nodes in wireless Networks. Hespanha and Bohacek talk about routing games in which an adversary tries to intersect data packets in a computer network [5]. Shuang-can et al. [6], propose a framework that applies two game-theoretic schemes for the economic deployment of intrusion detection agents. A game-theoretical framework that can model the interaction between the service provider and the attacker as an intrusion detection game was proposed by Kodialam and Lakshman [7]. Agah [8] and Basar [9], address the attack defense problem in a sensor network as a two-player non-cooperative, non-zero-sum game. Syverson suggests using stochastic games for reasoning and analysis [10]. Markovic and Reiher [11], present a taxonomy of Distributed Denial of Services attack and defense mechanisms to classify attacks and defense strategies. FRIARS cyber-defense decision system, which is capable of reacting autonomously to automated system attacks, is developed by McInerney et al. [12]. Marti et al. [13] propose an IDS scheme for MANET containing two different modules, viz. the Watchdog and the Path rater.

Extensive-form games are a broad type of game that can simulate the circumstances in many other areas. These areas include real-world settings such as security games [14,15], negotiation, cybersecurity games [16,17], recreational games [18] such as poker and billiards [19], and medical treatment [20,21]. Potential application scenarios also include robotics [22], such as multi-robot decision-making [23,24]. This paper describes a new method to find solutions in extensive games, and the basis of this method is regret minimization.

The concept of regret has received more and more attention in recent days. In 1950s, Hannan [25] gave an algorithm based on regret-minimizing for repeated two-player games. The stress of recent work on regret minimization has been laid on convergence rates and algorithmic efficiency. Kalai and Vempala [26] prove that Hannan's algorithm can be used to solve online linear optimization problems with regret approaching 0 at a rate $O(1/\sqrt{T})$, given access to an exact best-response oracle. Zinkevich [27] developed a regret-minimizing algorithm for online convex optimization problems. So-called bandit algorithms have also been developed [28], which achieve low regret even in the situation where the algorithm receives very limited information after each round of play. Kakade et al. [29] present a method in which an α -approximate best-response oracle can be used to find solutions to linear optimization problems.

In this paper, we describe a method to find solutions in extensive games. The basis of this method is regret minimization. Then we show how minimizing overall regret can help us make decisions in extensive games through a concrete case.

2. Nash Equilibria and Regret in Extensive Games

A finite extensive game with imperfect information has several components, listed as follows.

(1) There is a finite set N which contains the players.
(2) There is a finite set K that consists of all the possible results that the actions may lead to. $Z \subseteq K$ are the final results.

(3) For each player $i \in N$, there is a function $u_i: Z \rightarrow R$ that describes the final payoff for the player. If $N = \{1, 2\}$ and $u_1 + u_2 = 0$, then we call the game a zero-sum extensive game.

(4) For each player i there is a strategy profile σ , σ_i is the strategy that player i takes, σ_{-i} refers to all the strategies in σ except σ_i . N_i is the set of strategies for player i .

If every player in the game is able to recall the actions that they performed before, then we say that the game has perfect recall. In this paper, we will lay stress on zero-sum, finite extensive games that have perfect recall.

2.1. Strategies

Let $\tau^\sigma(k)$ be the probability that history k occurs if the players decide to act in the way as σ suggests. $\tau^\sigma(k)$ can be decomposed into $\tau^\sigma = \prod_{i \in N \cup \{C\}} \tau_i^\sigma(k)$, where $\tau_i^\sigma(k)$ is the probability that history h occurs if player i acts according to σ . Let $\tau_{-i}^\sigma(h)$ be all players' contribution except player i . For $I \subseteq K$, define $\tau^\sigma(I) = \sum_{h \in I} \tau^\sigma(k)$, as the probability of reaching a particular information set given σ , and $\tau_i^\sigma(I)$, $\tau_{-i}^\sigma(I)$ are defined in a similar way. The overall value to player i of a strategy profile is then the expected payoff of the final node, $\mu_i(\sigma) = \sum_{h \in Z} \mu_i(k) \tau_\sigma(k)$.

2.2. Nash Equilibrium

Traditionally, we use the concept of Nash equilibrium to find out the solution of a two-player extensive game. A Nash equilibrium is a strategy profile σ satisfying the following conditions:

$$\mu_1(\sigma) \geq \max_{\sigma'_1 \in N_1} \mu_1(\sigma'_1, \sigma_2), \quad (1)$$

$$\mu_2(\sigma) \geq \max_{\sigma'_2 \in N_2} \mu_2(\sigma_1, \sigma'_2). \quad (2)$$

An approximation of a Nash equilibrium or ϵ -Nash equilibrium is a strategy profile σ satisfying the following conditions:

$$\mu_1(\sigma) + \epsilon \geq \max_{\sigma'_1 \in N_1} \mu_1(\sigma'_1, \sigma_2), \quad (3)$$

$$\mu_2(\sigma) + \epsilon \geq \max_{\sigma'_2 \in N_2} \mu_2(\sigma_1, \sigma'_2). \quad (4)$$

2.3. Regret Minimization

The concept of regret has received more and more attention in recent days. To illustrate the definition of regret minimization, suppose that we are going to play an extensive game for several rounds. Let σ_i^t be the strategy that player i use on round t . The definition of the overall average regret of player i at time T is:

$$R_i^T = \frac{1}{T} \max_{\sigma_i^* \in N_i} \sum_{t=1}^T (\mu_i(\sigma_i^*, \sigma_{-i}^t) - \mu_i(\sigma_i^t)). \quad (5)$$

Regret minimization can be used as a method for choosing σ_i^t for player i if the overall average regret of player i tends to zero as t tends to infinity. Therefore, regret minimizing algorithms can be used as a method for finding a Nash equilibrium in extensive games. Moreover, the bounds of an algorithm on the overall average regret also bound the rate of convergence of the approximation.

In the next section, we will demonstrate how to use the concept of regret minimization to help players make decisions through a concrete case.

3. Case Study

In this section, we will apply the method of regret minimization to a practical extensive form game with incomplete information called Hearthstone.

3.1. Introduction of game rules

3.1.1 The game

First, we will give a brief introduction to the game. Hearthstone is a turn-based card game between two players. The aim of the game is to use the cards we have to destroy the opponent's hero, which means we need to reduce the health of the opponent's hero to zero. There are several ways to achieve this goal using different types of cards. The first way is to use minion cards to attack the opponent's hero. Each minion has an attack value and a health value. When a minion attacks the opponent's hero, it will deal damage to the opponent's hero, the amount of the damage is equal to its attack value. A minion can also be used to attack the opponent's minion, when a minion attacks another minion, both minions will deal damage to the other minion equal to its attack value. When the health value of a minion is reduced to zero or below, it will be destroyed. This paper uses a/b to describe a minion with attack value a and health value b . The second way is to use spells that can deal direct damage, spells with the description "deal x damage" can be targeted at the opponent's hero and reduce the health of the opponent's hero by x points. The third way is to use weapons. When we use a weapon, our hero will gain an attack equal to the attack value of the weapon, then we can use our hero to attack enemy hero or enemy minions. All these three kinds of cards have a cost value, which is called mana crystals. In each turn, the manas a player has is limited, so players need to decide which cards to use.

The game has three features: first, it is zero-summed, because only one player can win this game; second, it has imperfect information, since we do not know what card the opponent has in his hand; third, it is turn-based, two players take turns to make movements.

3.1.2 The case

In this paper, we will be focusing on a concrete case of the game. In this case, our hero has 7 health points, and we have a 1/1 minion that can attack. The opponent's hero has 6 health points, and he has a 1/1 minion and a special 2/3 minion with the description "whenever this minion takes damage, deal 2 damage to enemy hero". Now suppose that we have four mana crystals and three cards in our hand, the description of the three cards is listed as follows.

Table 1. Description of the cards we can use

Name	Type of the card	Mana Cost	Effect of the card
Mad Bomber	minion	2	Deal 3 damage randomly split between all other characters; summon a 3/2 minion

Frostbolt	spell	2	Deal 3 damage to a character
Arcane Explosion	spell	2	Deal 1 damage to all enemy minions

We need to notice that the effect of “mad bomber” is actually like this: deal 1 damage to a random character (except itself), repeated twice, and it won’t deal damage to a dying minion. For example, if the first 1 damage is dealt to our 1/1 minion, which is enough to destroy it, then the next 2 damage will not be dealt to this minion again.

3.2. Strategies

From section 3.1.2 we can learn that the choice we need to make is which two cards to use (and what target, if we use “frostbolt”), when and what target should we use our 1/1 minion to attack. After excluding improper movements, the main strategies we can take are listed in Table 2.

Table 2. Description of the six strategies

Strategies	Description of the strategy
Strategy1	We use the 1/1 minion to attack enemy hero, then use “frostbolt” targets at the 2/3 minion, and finally use “arcane explosion”. If we use this strategy, we will get a fixed result: our hero has 5 health, and we have a 1/1 minion; the opponent's hero has 5 health, and the opponent has no minion.
Strategy2	We use “frostbolt” targets at the 2/3 minion, then use our 1/1 minion to attack enemy hero, and finally use “mad bomber”.
Strategy3	We use “frostbolt” targets at the 2/3 minion, then use our 1/1 minion to attack enemy 1/1 minion, and finally use “mad bomber”.
Strategy4	We use “frostbolt” targets at the 2/3 minion, then use “mad bomber”, and finally use our 1/1 minion (if it is alive) to attack enemy 1/1 minion. (If that minion is alive, if not, attack enemy hero)
Strategy5	We use our 1/1 minion to attack enemy hero, then use “frostbolt” targets at the enemy hero, and finally use “mad bomber”. The benefit of this strategy is that we have the chance to destroy enemy hero (if mad bomber deals at least 2 damage to enemy hero), the probability of this is $P = \frac{1}{25} + \frac{2}{5} \times \frac{1}{16} + \frac{1}{125} + \frac{1}{5} \times \frac{2}{5} \times \frac{1}{4} + \frac{1}{125} = 10.1\%$. However, we also need to take a risk. The risk is that our hero might be easily destroyed in our opponent's turn, which means our opponent can destroy our hero simply by using the minions he has to attack our hero. The probability of this circumstance is $P = \frac{1}{5} \times \frac{1}{25} + \frac{1}{5} \times \frac{1}{16} + \frac{1}{5} \times (\frac{1}{20} + \frac{7}{25}) = 11.05\%$.
Strategy6	We use “frostbolt” targets at enemy hero, then use “mad bomber”, and finally use 1/1 to attack enemy hero. (If it is still alive and we can destroy enemy hero) If we choose this strategy, the probability that we can destroy enemy hero is $P = \frac{4}{125} + \frac{1}{5} \times \frac{1}{16} + \frac{2}{125} + \frac{1}{5} \times \frac{1}{5} \times \frac{1}{4} = 7.05\%$. The probability that our hero will be directly destroyed next turn is $P = \frac{1}{80} + \frac{1}{5} \times \frac{1}{25} + \frac{1}{5} \times (\frac{1}{20} + \frac{3}{25}) = 5.45\%$.

3.3. Regret minimization in this case

It’s not easy to tell which strategy is the most proper among these six strategies. Strategy 1 is the most conservative, and strategies 2,3,4 are rather conservative, although we might be able to destroy enemy hero if we take strategies 5 and 6, we also need to take the high risk of losing the game. In order to compare these strategies, we define two kinds of regret in this case and find out how they

change according to the number of cards our opponent has, and finally pick out the most proper strategy.

3.3.1 Definition

In our case, we define two kinds of regret. R_1 stands for the consequence that our opponent has enough direct damage (in his hand) to destroy our hero, $\mu_i(k) = -100$. R_2 stands for the difference between the upper limit of the strategy and the highest upper limit among all strategies. Since Strategy 5 has the highest upper limit, if we choose this strategy, we have a chance of 10.1% to destroy enemy hero. Therefore, we set the regret of this strategy to be 0, then we assess the upper limit of the rest of the strategies, compare them with the upper limit of Strategy 5, and set R_2 for them. The upper limit of the strategies and the amount of R_2 are listed in Table 3.

Table 3. The upper limit and R_2 of the strategies

Strategies	The upper limit of the strategy	R_2
Strategy1	Our hero has 5 health, and we have a 1/1 minion; the opponent's hero has 5 health, and the opponent has no minion.	-30
Strategy2	Our hero has 5 health, and we have a 1/1 minion and a 3/2 minion; the opponent's hero has 4 health, and the opponent has no minion.	-24
Strategy3	Our hero has 5 health, and we have a 3/2 minion; the opponent's hero has 3 health, and the opponent has no minion.	-26
Strategy4	Our hero has 5 health, and we have a 1/1 and a 3/2 minion; our opponent's hero has 5 health, and the opponent has no minion.	-24
Strategy5	10.1% to destroy enemy hero.	0
Strategy6	7.05% to destroy enemy hero.	-9

3.3.2 Calculation

Since R_2 is fixed for each strategy, all we need to calculate is R_1 . As we have defined, if our opponent has enough direct damage (in his hand) to destroy our hero, $\mu_i(k) = -100$, if not, $\mu_i(k) = 0$, so $\max_{\sigma_i^* \in N_i} \sum_{t=i}^T (\mu_i(\sigma_i^*, \sigma_{-i}^t)) = 0$, then $R_1 = \sum_{h \in Z} \mu_i(k) \tau^\sigma(k)$. Therefore, we need to calculate the probability that our opponent has enough direct damage to destroy our hero.

Let m be the remaining health of our hero, y be the direct damage that our opponent can deal to our hero, x be the number of cards enemy has.

First of all, we need to calculate in each strategy, the remaining health of our hero, which means the damage that our opponent needs to destroy our hero. The result is presented in Table 4.

Table 4. The possible remaining health of our hero

$S_i \setminus m$	0	1	2	3	4	5
S1	0	0	0	0	0	1
S2	0	0.0156	0.1111	0.2396	0.3889	0.2448
S3	0	0	0.125	0.375	0.375	0.125
S4	0	0	0.0799	0.2396	0.4201	0.2604
S5	0.1105	0.1475	0.2005	0.2629	0.1698	0.1088
S6	0.0545	0.1475	0.2325	0.2869	0.1698	0.1088

The rows in Table 4 represent the strategy we choose, the columns stand for the remaining health of our hero, and the number means the probability of this circumstance. For instance, the number in the third row and the second column means that if we choose strategy 2, the probability that our hero has one health point left is 1.56%. According to a statistic about the cards in Hearthstone, $y \sim N(x^{\frac{1}{2}}, x^{\frac{1}{3}})$. Then we get the relationship between the number of cards our opponent has and the direct damage that he can deal, as presented in Table 5.

Table 5. the possible direct damage enemy can deal

y\x	1	2	3	4	5
1	0.5	0.1587	0.0228	0.0013	0.0001
2	0.6293	0.3192	0.1038	0.0197	0.0022
3	0.695	0.4247	0.1894	0.0582	0.0116
4	0.7357	0.5	0.2643	0.1038	0.0294
5	0.7642	0.5516	0.3264	0.1515	0.0526
6	0.7881	0.5987	0.3821	0.1977	0.0808
7	0.8051	0.6331	0.4247	0.2389	0.1093
8	0.8186	0.6591	0.4641	0.2776	0.1386
9	0.8315	0.6844	0.5	0.3156	0.1685
10	0.8413	0.7054	0.5319	0.3483	0.1977

The rows in Table 5 represent the number of cards that the opponent has, the columns stand for the maximum direct damage that the opponent can deal, and the number means the probability of this circumstance. For instance, the number in the second row and the third column means that if the opponent has one card in his hand, then the probability that he can deal at least 2 damage to our hero is 15.87%. Then we can calculate the probability that our opponent has enough direct damage to destroy our hero given the strategy we take and the number of cards the opponent has, as presented in Table 6.

Table 6. The probability that our hero will be destroyed next turn

x\S _i	S1	S2	S3	S4	S5	S6
1	0.0001	0.0314	0.0289	0.0187	0.2223	0.1719
2	0.0022	0.0784	0.0865	0.0503	0.2982	0.2549
3	0.0116	0.1289	0.1503	0.1068	0.3591	0.3212
4	0.0294	0.1779	0.2042	0.1545	0.4096	0.3759
5	0.0526	0.2232	0.2547	0.1996	0.4511	0.4206
6	0.0808	0.267	0.3024	0.2435	0.4896	0.4619
7	0.1093	0.3043	0.3417	0.2812	0.5203	0.4948
8	0.1386	0.3391	0.3779	0.3166	0.5476	0.5239
9	0.1685	0.3728	0.4125	0.3509	0.5737	0.5516
10	0.1977	0.4028	0.4430	0.3816	0.5965	0.5759

The rows in Table 6 represent the number of cards the opponent has, the columns stand for the strategy we choose, and the number shows the probability that our hero will be destroyed next turn.

For instance, the number in the fourth row and the fifth column means that if the opponent has three cards in his hand and we choose strategy 4, then the probability that our hero will be destroyed next turn is 10.68%.

3.3.3 The overall regret of each strategy

Finally, we calculate the overall regret of each strategy, as presented in Table 7, and the changing trend of each strategy is presented in Figure 1. From the figure we can find that as the number of cards that the opponent has increases, the regret of every strategy increase, this is because the more cards opponent has, the higher the probability that he has enough damage to destroy our hero will be, leading to an increase in R_1 . The overall regret of strategy 1 increases much slower than the other strategies because if we choose this strategy, the remaining health of our hero will be fixed at the highest possible amount, so the probability that our hero will be destroyed next turn is much smaller. Therefore, although the upper limit of strategy 1 is lower than other strategies, leading to a higher R_2 in strategy 1, the robustness of this strategy ensures it a much lower level in R_1 as the number of cards the opponent has increases and consequently a lower overall regret than other strategies.

Table 7. the overall regret of each strategy

$x \backslash S_i$	S1	S2	S3	S4	S5	S6
1	-30.01	-27.14	-28.89	-25.87	-22.23	-26.29
2	-30.22	-31.84	-34.65	-29.03	-29.82	-34.49
3	-31.16	-36.89	-41.03	-34.68	-35.91	-41.12
4	-32.94	-41.79	-46.42	-39.45	-40.96	-46.59
5	-35.26	-46.32	-51.47	-43.96	-45.11	-51.06
6	-38.08	-50.7	-56.24	-48.35	-48.96	-55.19
7	-40.93	-54.43	-60.17	-52.12	-52.03	-58.48
8	-43.86	-57.91	-63.79	-55.66	-54.76	-61.39
9	-46.85	-61.28	-67.25	-59.09	-57.37	-64.16
10	-49.77	-64.28	-70.3	-62.16	-59.65	-66.59



Figure 1. The changes in the overall regret of each strategy

4. Conclusions

In this paper, we introduce the regret minimization methods for extensive-form games. We study the case of Hearthstone in order to demonstrate how we use the method of regret minimization can be used to make decisions by players. In this case, we need to find out the most proper strategy given the number of cards the opponent has. We calculate the overall regret of each strategy and reach the conclusion. From the perspective of regret minimization, if our opponent has only one card in his hand, Strategy 5 is the most proper; if our opponent has two cards in his hand, Strategy 4 is the most proper; if our opponent has more than three cards in his hand, Strategy 1 is the most proper.

References

- [1] Koller D. and Megiddo. N. (1992) The complexity of two-person zero-sum games in extensive form. *Games and Economic Behavior*, pages 528–552.
- [2] Hamilton S. N., Miller W. L., Ott A., and Saydjari O. S. (2002) Challenges in Applying Game Theory to the Domain of Information Warfare, *Proc. 4th Inf. Surviv. Work.*
- [3] Chakrabarti A. and Manimaran G. (2002) Internet Infrastructure Security: A Taxonomy, *IEEE Netw.*, no. December, pp. 13– 21.
- [4] Liu Y., Comaniciu C., and Man H. (2006) Modelling misbehaviour in ad hoc networks: a game theoretic approach for intrusion detection, *Int. J. Secur. Networks*, vol. 1, no. 3/4, p. 243.
- [5] Hespanha J. P. and Bohacek S. (2001) Preliminary Results in Routing Games, *Am. Control Conf. 2001. Proc. 2001. IEEE*, vol. 3, pp. 1904–1909.
- [6] Shuang-can Z., Chen-jun H., and Zhang W. (2009) Distributed intrusion detection system based on BP neural network, *Int. J. Secur. its Appl.*, vol. 8, no. 2, pp. 183–192.
- [7] Kodialam M. and Lakshman T. V., (2003) Detecting Network Intrusions via Sampling: A Game Theoretic Approach, *Infocom*, vol. 0, no. C.
- [8] Agah A., Das S. K., and Basu K. (2004) A game theory based approach for security in wireless sensor networks, in *IEEE International Conference on Performance, Computing, and Communications*, pp. 259–263.
- [9] Alpcan T. and Basar T. (2003) A game theoretic approach to decision and analysis in network intrusion detection, in *42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475)*, vol. 3, pp. 2595–2600.
- [10] Syverson, P. F. (1997) A different look at secure distributed computation. In *Proceedings 10th Computer Security Foundations Workshop*, *IEEE Comput. Soc. Press.*, pp. 109–115.
- [11] Mirkovic J. and Reiher P. (2004) A taxonomy of DDoS attack and DDoS defense mechanisms, *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 2, pp. 39–53.
- [12] McInerney J., Stubberud S., and Anwar S. (2001) Friars: a feedback control system for information assurance using a markov decision process, *Technol.*
- [13] Marti S., Giuli T., Lai K., and Baker M. (2000) Mitigating routing misbehavior in mobile ad hoc networks, *Proc. 6th Annu.*
- [14] Yee T., Lisý V., Bowling M. H. (2016) Monte Carlo Tree Search in Continuous Action Spaces with Execution Uncertainty[C]//IJCAI. pp. 690-697.
- [15] de Cote E. M., Stranders R., Basilico N., et al. (2013) Introducing alarms in adversarial patrolling games[C]//AAMAS. pp. 1275-1276.
- [16] Kim Y. S., DeBruhl B., Tague P. (2014) JADE: jamming-averse routing on cognitive radio mesh networks[C]//2014 IEEE Conference on Communications and Network Security. IEEE, pp. 21-28.
- [17] Chen L. C., Zhu Y., Papandreou G., et al. (2018) Encoder-decoder with atrous separable convolution for semantic image segmentation[C]//Proceedings of the European conference on computer vision (ECCV), pp. 801-818.
- [18] Sandholm T. (2010) The state of solving large incomplete-information games, and application to poker[J]. *Ai Magazine*, 31(4): 13-32.

- [19] Archibald C., Shoham Y. (2009) Modeling billiards games[C]//Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1, pp. 193-199.
- [20] Chen K., Bowling M. (2012) Tractable objectives for robust policy optimization[J]. Advances in Neural Information Processing Systems, pp. 25.
- [21] Sousa S., Brion R., Lintunen M, et al. (2015) Human breast cancer cells educate macrophages toward the M2 activation status[J]. Breast cancer research, 17(1): 1-14.
- [22] Chen, J., Xie, Z., Dames, P. (2022) The semantic PHD filter for multi-class target tracking: From theory to practice. Robotics and Autonomous Systems, 149, 103947.
- [23] Chen, J., Dames, P. (2020) Collision-free distributed multi-target tracking using teams of mobile robots with localization uncertainty. In 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 6968-6974). IEEE.
- [24] Chen, J., Dames, P. (2020, July) Distributed and collision-free coverage control of a team of mobile sensors using the convex uncertain voronoi diagram. In 2020 American Control Conference (ACC) (pp. 5307-5313). IEEE.
- [25] Hannan J. (1957) Approximation to Bayes risk in repeated play. In M. Dresher, A. Tucker, and P. Wolfe, editors, Contributions to the Theory of Games, volume III, pp. 97–139.
- [26] Kalai A. and Vempala S. (2003) Efficient algorithms for on-line optimization. In Proceedings of the The 16th Annual Conference on Learning Theory, pp. 26–40.
- [27] Zinkevich M. (2003) Online convex programming and generalized infinitesimal gradient ascent. In Proceedings of the 20th International Conference on Machine Learning, pp. 928–936.
- [28] Kleinberg R. (2006) Anytime algorithms for multi-armed bandit problems. In Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm, pp. 928–936.
- [29] Kakade S., Kalai A. T., and Ligett K. (2007) Playing games with approximation algorithms. In Proceedings of the 39th ACM Symposium on Theory of Computing (STOC).