COMPUTING TEAM-MAXMIN EQUILIBRIA IN ZERO-SUM MULTIPLAYER GAMES

YOUZHI ZHANG

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

NANYANG TECHNOLOGICAL UNIVERSITY

2020

COMPUTING TEAM-MAXMIN EQUILIBRIA IN ZERO-SUM MULTIPLAYER GAMES

YOUZHI ZHANG

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

A thesis submitted to the Nanyang Technological University

in partial fulfillment of the requirement for the degree of

Doctor of Philosophy

2020

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

Youzhi Zhang

February 22, 2020

....

Date Youzhi Zhang

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

February 22, 2020	[illegible]
....	....
Date	Bo An

Authorship Attribution Statement

This thesis contains materials from five papers published/submitted in the following peer-reviewed conferences where I was the first and/or corresponding author.

Chapter 3 is published as **Youzhi Zhang** and Bo An. Converging to team-maxmin equilibria in zero-sum multiplayer games. *Proceedings of the 37th International Conference on Machine Learning* (**ICML**), accepted, 2020. The contributions of the co-authors are as follows:

- I proposed the problem;
- I proposed the algorithms, conducted the experiments, and wrote the manuscript;
- Prof. Bo An provided insightful comments to the manuscript.

Chapter 4 is published as **Youzhi Zhang**, Bo An, Long Tran-Thanh, Zhen Wang, Jiarui Gan, Nicholas R. Jennings. Optimal escape interdiction on transportation networks. *Proceedings of the 26th International Joint Conference on Artificial Intelligence* (**IJCAI**), pp.3936-3944, 2017. The contributions of the co-authors are as follows:

- Prof. Bo An, Prof. Long Tran-Thanh, and Prof. Nicholas R. Jennings provided the initial project direction;
- Dr. Zhen Wang with the help of Mr. Jiarui Gan proposed the initial model and the initial algorithm and wrote the corresponding part of the initial manuscript;
- Prof. Bo An provided critical comments to the model;
- I proposed and implemented the advanced algorithms to make the work applicable and conducted the experiments;

- I wrote the manuscript;
- Prof. Bo An, Prof. Long Tran-Thanh, and Prof. Nicholas R. Jennings provided comments to the manuscript;
- I and Prof. Bo An revised the paper.

Chapter 5 is published as **Youzhi Zhang** and Bo An. Computing team-maxmin equilibria in zero-sum multiplayer extensive-form games. *Proceedings of the 34th AAAI Conference on Artificial Intelligence* (**AAAI**), accepted, 2020. The contributions of the co-authors are as follows:

- I proposed the problem;
- I proposed the algorithms, conducted the experiments, and wrote the manuscript;
- Prof. Bo An provided insightful comments to the manuscript.

Chapter 6 is submitted as **Youzhi Zhang** and Bo An. Computing *ex ante* coordinated team-maxmin equilibria in zero-sum multiplayer extensive-form games. *Under review by the 34th Conference on Neural Information Processing Systems (**NeurIPS**)*, 2020. The contributions of the co-authors are as follows:

- I proposed the problem;
- I proposed the algorithms, conducted the experiments, and wrote the manuscript;
- Prof. Bo An provided insightful comments to the manuscript.

Chapter 7 is published as **Youzhi Zhang**, Qingyu Guo, Bo An, Long Tran-Thanh, Nicholas R. Jennings. Optimal interdiction of urban criminals with the aid of real-time information. *Proceedings of the 33rd AAAI Conference on Artificial Intelligence* (**AAAI**), pp.1262-1269, 2019. The contributions of the co-authors are as follows:

- I proposed the problem and the model;

- Prof. Bo An and Dr. Qingyu Guo provided critical comments to the model and I justified the model;
- I proposed the algorithms, conducted the experiments, and wrote the manuscript;
- Prof. Bo An, Prof. Long Tran-Thanh, and Prof. Nicholas R. Jennings provided insightful comments to the manuscript;
- I and Dr. Qingyu Guo revised the manuscript.

February 22, 2020 Youzhi Zhang

....

Date Youzhi Zhang

Abstract

Efficiently computing Nash Equilibria (NEs) for multiplayer games is still an open challenge in computational game theory. In fact, it is not only hard to compute NEs in multiplayer games, but also hard for players independently choosing strategies and then forming an NE because NEs are not exchangeable. This thesis focuses on computing Team-Maxmin Equilibria (TMEs), which was introduced as a solution concept for zero-sum multiplayer games where players in a team having the same utility function play against an adversary. The Correlated TME (CTME) is a variant of the TME, where team players can synchronize (correlate) their strategies by exploiting a mediator recommending actions to them through communication. According to different forms of communication, the CTME in extensive-form games includes: 1) the TME with Coordination device (TMECor) for the situation where team players can communicate their actions only before the play of the game; and 2) the TME with Communication device (TMECom) for the situation where team players can communicate their actions before the play of the game and during the game's execution. The TME has some fascinating properties, e.g., it is unique in general, which can avoid the equilibrium selection problem. Moreover, TMEs can capture many realistic scenarios, including: 1) a team of players play against a target player in poker games; and 2) multiple defense resources schedule and patrol against the adversary in security games. Unfortunately, existing algorithms are inefficient to compute TMEs in large games. Therefore, in this thesis, we develop efficient algorithms to compute TMEs in normal-form and extensive-form games, including general algorithms and domain-specific algorithms for applications.

First, we develop a novel incremental strategy generation algorithm to compute TMEs in normal-form games efficiently, which is the first incremental strategy generation algorithm guaranteeing to converge to a TME. Second, we apply the TME to a novel escape interdiction game model capturing the interaction between an escaping adversary and a team of multiple defense resources with correlated time-dependent strategies on transportation networks, and then we efficiently compute CTMEs in normal-form games for optimal escape interdiction on such networks with a domain-specific incremental s-trategy generation algorithm. Third, to efficiently solve the non-convex program for finding TMEs in extensive-form games, we develop a novel global optimization technique, the associated recursive asynchronous multiparametric disaggregation technique, to approximate multilinear terms in the non-convex program, and then we develop a corresponding novel iterative algorithm to compute TMEs within any given accuracy efficiently. Fourth, to efficiently compute TMEsCor in extensive-form games, we develop an incremental strategy generation algorithm converging within finite iterations in the infinite strategy space, where we develop a novel global optimization technique, the associated representation technique, to exactly represent multilinear terms in the best response oracle. Fifth, we apply the TME to a novel network pursuit game model capturing the interaction between an escaping adversary and a team of multiple defense resources who can communicate during the game's execution on urban networks, and then we efficiently compute TMEsCom in extensive-form game for optimal interdiction of urban criminals with the aid of real-time information by a domain-specific incremental strategy generation algorithm. Finally, experimental results show that our general algorithms are orders of magnitude faster than prior state-of-the-art algorithms in large games, and our domain-specific algorithms can scale up to realistic problem sizes with hundreds of nodes on networks including the real network of Manhattan.

Acknowledgements

Reaching the end of a long journey, I would like to express a great sense of gratitude to people who helped and inspired me in the past three and a half years. During this period, there are too many sleepless nights when I faced difficulties in research. However, I cannot remember how many exciting moments I had when I conquered those difficulties. I have enjoyed this wonderful journey as a Ph.D. student with the help of many excellent people.

First of all, I want to thank my supervisor Prof. Bo An. I cannot reach this point without your enormous help and instructions. Thank you very much for giving me this great opportunity to pursue my Ph.D. in your group. Since the beginning of this journey, I have been significantly impressed by your professional enthusiasm for research, which I will continue striving for. Thanks very much for your patience and helpful guidance during this journey. Your insightful comments always deepen my understanding of the specific research problem, and your tremendous help has improved my writing style. You have reshaped my understanding of research, e.g., we should do the research with fundamental contributions. I am also inspired by your commitment to study real-world problems that benefit society. You not only have set an example on how to do research for me, but also taught me about how to be responsible and professional, and a lot of others. I will be forever thankful for your supervision. I hope the relationships and connections built in your group will grow stronger over time and last forever.

I would like to thank my Thesis Advisory Committee (TAC) members: Prof. Yang Liu and Prof. Zhiwei Wang. The discussion with you helped me a lot for improving my research. I will forever remember your valuable guidance and suggestions. I would like to thank Prof. Xiaohui Bei who taught my first course on algorithms, which has inspired me in my research. I also would like to thank many professors who taught me courses during this journey, which have helped me a lot.

During this long journey, I am grateful to have the privilege to collaborate with many extraordinary excellent researchers: Dr. Qingyu Guo, Dr. Zhen Wang, Jiarui Gan, Prof. Long Tran-Thanh, and Prof. Nicholas R. Jennings. I thank you for your guidance and insights in those projects, as well as the great patience of mentoring.

It was fortunate for me to share my Ph.D. experience with a group of talented people: Haipeng Chen, Yanhai Xiong, Qingyu Guo, Mengchen Zhao, Zhen Wang, Jiarui Gan, Jiuchuan Jiang, Martin Strobel, Wanyuan Wang, Lei Feng, Xu He, Aye Phyu, Hongxin Wei, Wei Qiu, Rundong Wang, Jakub Černý, Runsheng Yu, Renchi Zhang, Jianfeng Li, Yanchen Deng, Shuxin Li, Wanqi Xu, and Shenggong Ji. Thank you all for those moments of laughs, happiness, and excitement, and your generous help when I needed it. I will forever remember the moments for fighting together for deadlines, having internal group meetings, going hiking, and enjoying the group dinner with you. Thank you for calling me Chair on research.

I want to thank many of my friends: Yuhao Lu, Hao Su, Yanrong Xie, Xiaoyun Mo, Renchi Yang, Xiaoshi Zhong, Qiang Zhou, and so on.

Lastly but most importantly, I want to thank my parents for your unconditional love and supports.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Game theory is an important tool to model the interaction between agents. Now researchers have achieved many results for two-player games, e.g., computing Nash Equilibria (NEs) for zero-sum games [1] via linear programs [2–4] and many scalable algorithms, e.g., the double oracle algorithm [5] and the counterfactual regret minimization algorithm [6], and computing Stackelberg equilibria [7]. Based on these results, researchers have successfully applied game theory to many domains, e.g., improving the security for people and wildlife in security games [8] and defeating top human professionals in poker games [9]. However, researchers have achieved fewer results for multiplayer games except for games having very special structures, e.g., polymatrix games [10] and congestion games [4] or algorithms having no theoretical guarantee [11]. In fact, the hardness to compute NEs (it is PPAD-complete even for zero-sum three-player games [12]) and the equilibrium selection problem [11] (it is hard for players independently choosing strategies and then forming an NE because NEs are not exchangeable (see Figure 1.1)) make them remain open challenges for computing and applying NEs in multiplayer games.

This thesis focuses on computing Team-Maxmin Equilibria (TMEs) [13–16], which is an important solution concept for zero-sum multiplayer games where players in a team having the same utility function play against an adversary. The Correlated TME (CTME) is a variant of the TME, where team members can synchronize (correlate) their

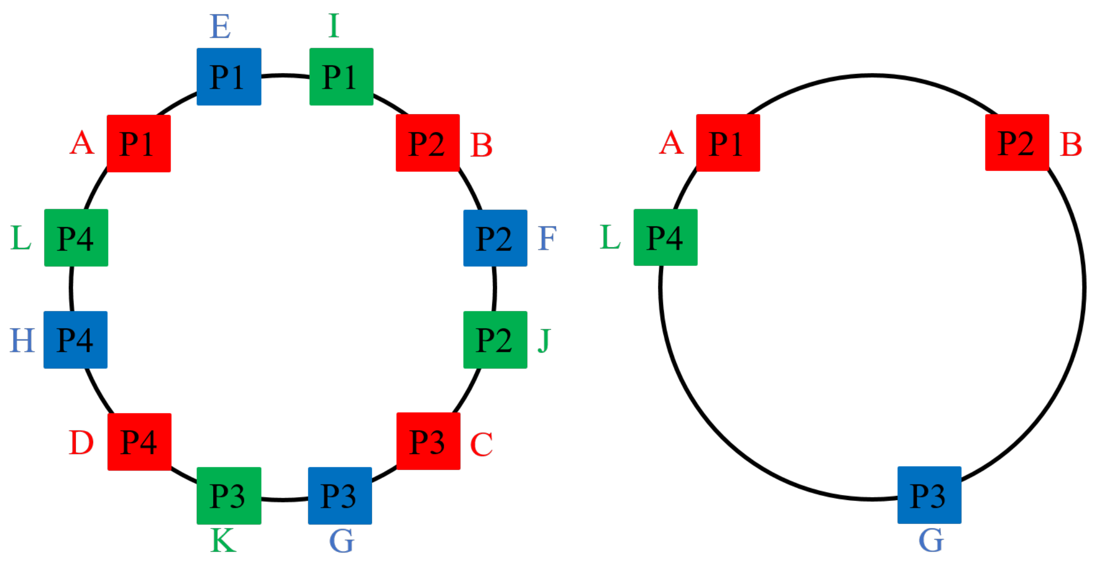


FIGURE 1.1: The equilibrium selection problem. In zero-sum two-player games, NEs are exchangeable. For example, if (A, B) and (C, D) are both NEs, so does (A, D). However, in multiplayer games, NEs are not exchangeable. This property of NEs in multiplayer games causes the equilibrium selection problem. We can illustrate this problem by using the Lemonade Stand Game [11], where players simultaneously select a point on a circle, and every player tries to stay as far away from other players as possible on the circle. In each NE of this game, players are spaced uniformly around the circle. Then we can have three different NEs for four players shown on the left circle: (A, B, C, D), (E, F, G, H), and (I, J, K, L). However, the joint strategy (A, B, G, L) independently selected by players is not an NE, as shown on the right circle.

strategies by exploiting a mediator recommending actions to them through communication. According to different forms of communication, CTMEs in extensive-form games include: 1) TMEs with Coordination device (TMEsCor) for the situation where team players can communicate and correlate their actions only before the play (*ex ante* coordination); and 2) TMEs with Communication device (TMEsCom) for the situation where team players can communicate and correlate their actions before the play of the game and during the game's execution. A TME is an NE maximizing the team's utility and always exists. More importantly, the TME is unique in general, and then it avoids the equilibrium selection problem. In addition, TMEs can capture many realistic scenarios. For example, in multiplayer poker games, all except one of them may form a team and play against the target player, i.e., they share the same goal but take actions independently (e.g., the setting of poker games in Brown and Sandholm (2019) avoids that players can communicate to correlate their actions). Especially, in the car game

Bridge, when the game reaches the phase of the play of the hand, two defenders, forming a team but playing cards independently due to the rule, play against the declarer. In addition to recreational applications, in security games, the US Coast Guard and other different police departments schedule and patrol independently against the attacker at major ports such as the New York port [17]. Specially, to keep the safety of New York, the Patrol Services Bureau in the New York City Police Department (NYPD) has 77 police precincts [18], and each precinct is divided into four or five fully-staffed sectors by Neighborhood Policing recently [19]. They maintain "sector integrity": officers in different sectors work independently to keep the safety of their sectors, except in precinct-wide emergencies [19]. On the other hand, there are many cases that the defender with multiple resources takes correlated actions against an adversary in security games, especially in network security games [20–23].

However, it is still challenging to compute a TME, which is FNP-hard [24]. Moreover, a TME is only solved via a non-convex program [13] with global optimization techniques [14, 15], which makes it hard to solve large games. It is also very hard to compute CTMEs [14, 15] due to the combinatorial action space, i.e., the Cartesian product of the action spaces of team members [14, 15]. Existing algorithms are inefficient to compute TMEs in large games. Therefore, in this thesis, we develop efficient algorithms to compute TMEs in normal-form and extensive-form games. Our contributions include general algorithms and applications with domain-specific algorithms. Now we show the relation between the solution concept TME an its variants and then present a brief review of the contributions we made in this thesis.

Now we use a scenario to illustrate the difference between the TME and its variants, as shown in Table 1.1. This scenario is on urban security, where police officers need to respond quickly to catch the criminal after a crime event happens. Actually, this is a game between a team of police officers and an adversary, where the adversary picks one escape path to the outside world while each police officer chooses intersections to interdict the adversary. To design the optimal interdiction plan in this game, we may need to consider different situations. If there is no real-time information available during the play of the game, we can model the game as a normal-form game. If team players cannot communicate, we can use the TME in normal-form games as the solution

Move	Communication	Solution Concept	Algorithm
Simultaneous moves (Normal-form games)	No communication between team players	TME	Chapter 3
	Communication between team players	CTME	Chapter 4
Sequential moves (Extensive-form games)	No communication between team players	TME	Chapter 5
	Communication between team players before the game play	TMECor	Chapter 6
	Communication between team players before and during the game play	TMECom	Chapter 7

TABLE 1.1: The relation between the TME and its variants, and the corresponding structure of the thesis.

concept. In Chapter 3 of the thesis, we propose an efficient algorithm to compute it for general games. If team players can communicate and then correlate their actions, we use the CTME as the solution concept to exploit the coordination among team players. In Chapter 4 of the thesis, we propose a novel escape interdiction game model for a real-world problem and then propose an efficient algorithm to compute the corresponding CTME for this specific game. Nowadays, novel pursuit devices can provide the police with the real-time location information of the criminal, for example, UAVs and camera systems. With real-time information, the police can dynamically adjust the moves to interdict the attacker. To consider this real-time information, we can model the game as an extensive-form game. If there is no communication between team players, we use the TME in extensive-form games as the solution concept. In Chapter 5 of the thesis, we propose an efficient algorithm to compute it for general games. If team players can communicate before the play of the game, we use the TMECor as the solution concept to exploit the *ex ante* coordination among team players. In Chapter 6 of the thesis, we propose an efficient algorithm to compute it for general games. If team players can communicate before and during the play of the game, we use the TMECom as the solution concept to exploit the real-time coordination among team players. In Chapter 7 of the thesis, we propose a novel network pursuit game model for a real-world problem and propose an efficient algorithm to compute a TMECom for this specific game.

Overall, Chapter 3–4 study the TME computation and its application in normal-form games, and Chapter 5–7 study the TME computation and its application in extensive-form games. In terms of algorithms, this thesis proposes the first incremental strategy generation algorithm guaranteeing to converge to a TME in Chapter 3. Then this thesis proposes various novel components for the incremental strategy generation algorithm to improve scalability: 1) the component on how to initialize the strategies in the restricted game in Chapter 3; 2) the component on how to efficiently solve the restricted game in Chapter 5; and 3) several best response oracles on how to efficiently compute the best response in various scenarios in Chapters 4, 6 and 7. The algorithms of this thesis are supported by theoretical and experimental results, followed by real-world applications where more specific restrictions or information is available. That is, the first contribution proposes general algorithms in normal-form games, while the second contribution is an application with domain-specific algorithms in normal-form games; and the third and fourth contributions propose general algorithms in extensive-form games, while the fifth contribution is an application with domain-specific algorithms in extensive-form games. Experiments show that our general algorithms are orders of magnitudes faster than baselines. In addition, our algorithms for applications are tested on real-world datasets, which shows that our algorithms can scale up to realistic problem sizes with hundreds of nodes on networks, including the real network of Manhattan.

1.1 Computing Team-Maxmin Equilibria in Normal-Form Games

The first part of this thesis focuses on computing TMEs in normal-form games. Note that each extensive-form game can be represented by a normal-form game, so the methods in this part can also be applied to extensive-form games.

1.1.1 Converging to Team-Maxmin Equilibria

It is challenging to compute a TME, which is FNP-hard [24]. Moreover, a TME is only solved via a non-convex program [13] with global optimization techniques [16], which makes it hard to solve large games. In addition, it is impossible to apply this approach when we cannot represent the problem via the game matrix due to the large strategy space. For example, in a network security game on a fully connected network with 190 edges and 20 nodes, the number of possible adversary pure strategies (paths) without any cycle is about 6.6^{18} [20], which means that the memory cost for enumerating all pure strategies would be prohibitive. In two-player games, the Incremental Strategy Generation (ISG) algorithm (including the double oracle algorithm, column generation, and so on) [5, 8, 20] is an efficient approach to avoid enumerating all pure strategies: It computes an equilibrium in a game with restricted strategy spaces for players, and then iteratively expands players' strategy spaces. ISG can converge to an NE in zero-sum two-player games. However, the study of ISG for computing TMEs is completely unexplored. We know that ISG terminates when oracles in ISG cannot find better strategies than the equilibrium strategies in the restricted game, which is consistent with the definition of NEs. Unfortunately, NEs in multiplayer games, unlike those in two-player games, are not exchangeable and may give different utilities to the team. Therefore, if ISG is used to compute TMEs, even it can converge to an NE, it may cause a loss to the team.

To fill this gap, we first study the properties of ISG for multiplayer games, showing that ISG converges to an NE but may not converge to a TME, and it can cause an arbitrarily large loss to the team. Second, we design an ISG variant (ISGT) by exploiting that a TME is an NE maximizing the team's utility and show that ISGT converges to a TME and the impossibility of relaxing conditions in ISGT. Third, to further improve the scalability, we design an ISGT variant (CISGT) by using the strategy space for computing an equilibrium that is close to TME but is easier to be computed as the initial strategy space of ISGT. Finally, extensive experimental results show that CISGT is orders of magnitude faster than ISGT and the state-of-the-art algorithm to compute TMEs in large games.

1.1.2 Computing Correlated Team-Maxmin Equilibria

The CTME is a solution concept capturing scenarios where players in a team having the same utility function play against an adversary by synchronizing their actions. That is, team members can decide and execute actions jointly. A CTME is an NE in zero-sum multiplayer games and has the properties of NEs in zero-sum two-player games (i.e., a CTME is equivalent to an NE in zero-sum two-player games where the team is treated as a single player). CTMEs can capture many real-world scenarios, especially in network security games [20, 21]. This work uses the CTME as the solution concept for the novel escape interdiction game model that captures the interaction between an escaping adversary (attacker) and a team of multiple defense resources (defender) with time-dependent strategies on transportation networks.

Preventing crimes or terrorist attacks in urban areas is challenging, since the number of potential targets in cities and large towns is huge. For example, the large number of bank branches, especially those offering many escape routes, could be highly profitable targets for bank robbers. In case of such an event, e.g., bank robbery or terrorist attack on buildings, it is critical for police officers to respond quickly and catch the attacker on his escape route. In this work, we aim to help law enforcement agencies (defender) by efficiently scheduling security resources to interdict the escaping attacker.

It is a significant challenge to develop an effective response plan given the limited security resources and the attacker's strategic actions. Given the initial locations of the security resources, the defender needs to consider traffic-dependent travel time in planning the schedules. In addition, one defender resource may capture the attacker at different intersections on his possible escape routes. Therefore, to make the best use of her limited resources, the defender needs to dynamically relocate them among potential intersections during the event. Moreover, the attacker may strategically choose his escape path and dynamically change his driving speed on different roads along the path to avoid being captured, which makes the attacker's strategy space continuous.

To conquer this challenge, we introduce a novel Escape Interdiction Game (EIG) to model time-dependent strategies for both players, where the defender chooses a sequence of intersections for each police officer to protect, while the attacker chooses an escape path and his travel time on different edges along the path. However, due to these extensions, both the defender strategy space and the continuous attacker strategy space suffer an exponential growth. These lead to major challenges in the computation of the optimal solution, as we will prove this problem is NP-hard. In particular, state-of-the-art security game solutions, such as the current double oracle type algorithms [25, 26], can only be applied to our problem at very small scales. To overcome this computational challenge, we first propose our solution, *EIGS*, which incorporates the following key components: 1) a new double oracle structure to avoid calling the hard oracle, i.e., the best response attacker oracle, frequently; 2) novel mixed-integer linear programming (MILP) formulations to compute the best response strategies for both players; and 3) a greedy algorithm and an efficient modified Dijkstra algorithm to efficiently generate improving strategies. To further improve the scalability to solve large-scale scenarios with hundreds of intersections, we introduce a novel defender algorithm *RedAD* that reduces the strategy space by deploying each defender resource to protect only one node, assuming that the attacker travels with maximum speed, and using a novel algorithm to contract the graph before using the MILP to compute the attacker response strategy. We conduct extensive experiments showing that *EIGS* can efficiently obtain a robust solution significantly outperforming existing approaches in problems of related small scales and *RedAD* can scale up to realistic-sized transportation networks of hundreds of intersections with good solution quality.

1.2 Computing Team-Maxmin Equilibria in Extensive-Form Games

The second part of this thesis focuses on computing TMEs in extensive-form games. Note that we exploit game tree structure in extensive-form games to design efficient algorithms.

1.2.1 Computing Team-Maxmin Equilibria

It is challenging to compute a TME in extensive-form games, which is FNP-hard and formulated as a non-convex programming problem [15] even each player's (sequence-form) strategy space is linear in the size of extensive-form games. The current approach [15] to compute this TME is only to solve the non-convex problem directly by using the state-of-the-art global optimization solver BARON [27]. Unfortunately, BARON does not scale well [15]. On the other hand, in extensive-form games, some efficient algorithms [15, 28] are proposed to compute Correlated-Team Maxmin Equilibria (CTMEs), where team players correlate their strategies. However, if team players transform a CTME into the mixed strategy profile, they may obtain an arbitrarily large loss [14]. Therefore, the study of efficiently finding TMEs within any given accuracy, especially in extensive-form games, is almost completely unexplored. We then develop very scalable algorithms to compute TMEs.

We first study the inefficiency caused by computing a CTME and then transforming it into the mixed strategy profile of the team and theoretically show that this inefficiency can be arbitrarily large in extensive-form games. Second, to efficiently solve the non-convex program for finding TMEs directly, we develop the Associated Recursive Asynchronous Multiparametric Disaggregation Technique (ARAMDT) to approximate multilinear terms in the program by using a digit-wise discretization of one variable in the term and then transform the program into a mixed-integer linear program (MILP) with two novel techniques: 1) an asynchronous precision method, where we can use different precision levels to approximate these terms even they include the same discretized variable to reduce the number of constraints and variables for approximation; and 2) an associated constraint method, where we exploit the relation between these terms to reduce the feasible solution space of the MILP in the ARAMDT. To our best knowledge, neither of both techniques has been considered in the literature of global optimization. Third, we develop a novel algorithm to efficiently compute TMEs within any given accuracy based on the ARAMDT, which uses novel techniques to iteratively increase the precision levels for part of non-linear terms. Finally, we evaluate our algorithm by experiments showing that our algorithm is dramatically faster than baselines.

1.2.2 Computing Team-Maxmin Equilibria with Coordination Device

The TMECor [15] is a solution concept capturing scenarios where a team of players with *ex ante* coordination and the same utility function play against an adversary. That is, all team players can communicate only before the play through a coordination device. More specifically, team members can discuss and agree on tactics before the game starts, but they cannot communicate anymore during the game. A TMECor is an NE in zero-sum multiplayer extensive-form games and has the properties of NEs in zero-sum two-player games (i.e., a TMECor is equivalent to an NE in zero-sum two-player games). TMEsCor can capture many real-world scenarios, especially in network security games [20, 21]. For example, in multiplayer poker games, a team may play against the adversary player, but they cannot communicate and discuss during the game due to the rule. Especially, in Bridge, when the game reaches the phase of the play of the hand, two defenders, forming a team play against the declarer. In addition, in the war, colluders may not have time or means to communicate during the battle [28].

However, it is challenging to compute a TMECor, which is FNP-hard [15]. A T-MECor can be computed by solving a linear program [15], where the team plays joint normal-form strategies for all team members, but each member's normal-form strategy space is exponential in the size of the game tree. To compute TMEsCor efficiently, a column generation algorithm is proposed [15], whose most important component, the Best Response (BR) oracle computing the best response for the team against the adversary strategy, is formulated as a mixed-integer linear program. In this BR, the number of integer variables is equal to $|L|$ (L is the set of terminal nodes in a game), which is extremely large in large games. For example, $|L|$ is about 10^6 in the three-player Leduc Hold'em Poker games with four ranks. The large number of integer variables makes the algorithm inefficient to compute the best response and then makes it inefficient to compute TMEsCor in large games (it can compute TMEsCor only for very small three-player Kuhn poker games). Moreover, Farina et al. (2018) develop a realization-form strategy representation and then propose a fictitious-play based algorithm to compute approximate TMEsCor ($\epsilon\Delta_u$-TMEsCor, and Δ_u is the difference between the maximum and minimum possible utility). However, their algorithm is inefficient to compute ϵ-TMEsCor with small ϵ, let alone exact TMEsCor, e.g., it cannot converge to an $\epsilon\Delta_u$-TMECor with $\epsilon = 0.0004$ within 100 hours for the smallest three-player Kuhn poker game in our experiments. In addition, their representation focuses on the probability distribution on terminal nodes of the game tree, which forbids players to take their realization-form strategy from the root of the game tree. To make it executable, we need to reconstruct a normal-form strategy from this realization-form strategy. Unfortunately, it is not easy to do that in large games. For example, an existing reconstruction algorithm (Algorithm 2 in Celli et al. (2019)) runs in time $O(|L|^2)$, but, as we mentioned above, $|L|$ could be 10^6. To avoid such a cumbersome reconstruction algorithm, we propose a novel hybrid-form strategy representation for the team and develop a dramatically faster algorithm to compute TMEsCor based on it.

To compute TMEsCor efficiently, we first propose a novel hybrid-form strategy representation for the team, where one team member takes sequence-form strategies while other team members take normal-form strategies. Second, based on our hybrid-form strategies, we develop our column generation algorithm with two novel features: 1) we provide a finite upper bound for the number of iterations where our algorithm terminates even though the strategy space is infinite; and 2) we design a novel BR for the team, which involves multilinear terms representing reaching probabilities for terminal nodes to reduce the number of variables. Third, we develop our novel associated representation technique for solving our multilinear BR, which has two novel features: 1) exactly representing multilinear terms by linear constraints and 2) efficiently generating associated constraints for the equivalence relation between multilinear terms in the game tree. Finally, extensive experiments show that our algorithm is several orders of magnitude faster than state-of-the-art algorithms in large games. Thus, this work shows that, to compute equilibria, the approach formulating the problem as a multilinear program with global optimization techniques can be dramatically faster than the approach immediately formulating it as a linear program.

1.2.3 Computing Team-Maxmin Equilibria with Communication Device

The TMECom [15] is a solution concept capturing scenarios where a team of players with a communication device (similar to a mediator) and the same utility function play against an adversary. That is, all team players can communicate before and during the play through a communication device, which receives the information that the team members observed and sends recommendations related to the action to play at each information set for each team member. More specifically, team members can discuss and agree on tactics before the game starts, and they can communicate to receive recommendations during the game. A TMECom is an NE in zero-sum multiplayer extensive-form games and has the properties of NEs in zero-sum two-player games (i.e., a TMECom is equivalent to an NE in zero-sum two-player games). TMEsCom can capture many real-world scenarios, especially in network security games [20–22]. Here, we use the T-MECom as the solution concept for the novel network pursuit game model that captures the interaction between an escaping adversary and a team of multiple defense resources (defender) with real-time information available on urban networks.

In 2016, there were nearly 5.36 million violent crimes across the United States, most of which happened in the urban and suburban cities [30]. Considering just the commercial banks, 4,185 robberies happened [31]. When an urban crime occurs, the top priority of law enforcement officers is to dispatch the limited security resources to capture the criminal. Nowadays, novel pursuit devices such as the StarChase GPS-based system [32], UAVs and helicopters can provide the police with the real-time location of the criminal. Equipped with the up to date information, the police can dynamically adjust the resource allocation plan to interdict the adversary.

Unfortunately, existing work on resource allocation in network security ignores the real-time information about the adversary's location [20, 33, 34]. Specifically, in existing urban network security games (NSG), the defender usually deploys a time-independent strategy without considering the dynamic relocation of security resources

[20]. Although a time-dependent policy is discussed in [22], they do not take the real-time adversarial information into consideration, which can cause a huge loss of effectiveness as we will show later. Other branches of related work also suffer from similar myopia, including pursuit-evasion games [35] and patrolling security games [36].

To incorporate such real-time information, we model the urban network security problem as a zero-sum *NEtwork purSuiT game (NEST)*. In NEST, the adversary picks one escape path to the outside world, while the defender, tracking the adversary's previous moves, decides on the relocation of her security resources (modeled as a finite-horizon Markov Decision Process (MDP)). Specifically, in the MDP, each state includes information about the adversary's previous moves and the locations of defender resources, and each action is the next location of the defender's resources. This results in a game with a combinatorial action space of each state in the MDP for the defender, where the defender has multiple resources with several actions each. For example, for just ten resources with four actions for each, there are more than one million joint actions of resources in a given state. This large strategy space makes the existing exact solution approaches unable to solve NEST efficiently. These approaches include: i) the approach for the imperfect recall game, i.e., normal-form game with sequential strategies (NFGSS), because their incremental strategy generation (*ISG*) algorithm does not set bounds to cut branches in its best-response (*BR*) oracle that uses a depth-first search through the whole state space to compute the best response [37]; and ii) the approach for the perfect recall game [3] because it will lead to a significantly larger strategy space than the imperfect recall approach (see the discussion in [37]), and the bounds are very loose in their branch-and-bound *BR* [38].

In this context, after modeling NEST as an imperfect-recall game to reduce the s-trategy space, this work makes four additional key contributions. First, we show that the ignorance of real-time information in existing work can lead to an arbitrarily large loss of efficiency. Second, we show that solving NEST is NP-hard. Third, after transforming the non-convex program of solving NEST to a linear program (*LP*), we propose our *ISG* with the following novelty: (i) novel pruning techniques in our *BR*, including: (1) providing tight lower and upper bounds by exploiting the combinatorial structure of each state's action space; and (2) handling imperfect recall by developing an effective lower bound transition method that causes an evaluated state to be reevaluated only if a prefix state with a smaller lower bound transits to it; and (ii) novel techniques to speed up our *ISG* to solve extremely large-scale problems, which include: (1) mapping the computed defender strategy by *BR* between similar subgames to speed up *BR*, which also speeds up *LP* if this strategy is optimal; and (2) adding multiple best response strategies to the restricted game at one iteration against different sampled adversary strategies to reduce the number of iterations for convergence. Finally, experiments show that our approach can scale up to problem sizes with hundreds of nodes on networks including the real network of Manhattan. Thus, for the first time, this work shows that *ISG* techniques can also be scaled for dynamic games when best-response oracles are tuned with domain-specific pruning techniques, and other domain-specific improvements are employed.

1.3 Thesis Overview

The organization of this thesis is as follows. Chapter 2 discusses the background materials with the related work of this thesis. Chapter 3 considers the problem of converging to TMEs in normal-form games by the incremental strategy generation algorithm. Chapter 4 focuses on computing CTMEs in normal-form games for optimal escape interdiction on transportation networks. Chapter 5 focuses on computing TMEs in extensive-form games. Chapter 6 focuses on computing TMEsCor in extensive-form games. Chapter 7 focuses on computing TMEsCom in extensive-form games for optimal interdiction of urban criminals with the aid of real-time information. Chapter 8 summarizes this thesis and provides possible future directions.

Chapter 2

Background

This chapter reviews the background for this thesis. We discuss solution concepts in game theory and the general scalable approaches to compute them. Then we review existing research that is relevant to this thesis

2.1 Game Theory

Game theory is an effective mathematical model in regard to agents' strategy selection in the interactive environment [4]. When players move simultaneously, the corresponding game representation is normal-form games, and when players move sequentially, the corresponding game representation is extensive-form games. Now we introduce some notations on normal-form games and extensive-form games with some corresponding solution concepts.

2.1.1 Normal-Form Games

A normal-form game G [4] is a tuple (N, A, u) where: $N = \{1, \dots, n\}$ is a set of players, $A = \times_{i \in N} A_i$ is a set of joint actions with that A_i is a finite set of player i's actions (pure strategies) with $a_i \in A_i$, and $u = (u_1, \dots, u_n)$ is a set of players' utility functions with that $u_i : A \to \mathbb{R}$ is player i's utility function. $X = \times_{i \in N} X_i$ is the set of mixed strategy profiles with that $X_i = \Delta(A_i)$ is the set of player i's mixed strategy.

For each $x_i \in X_i$ and $a_i \in A_i$, $x_i(a_i)$ is the probability that action a_i is played, $\underline{A}_{i,x_i} = \{a_i \mid x_i(a_i) > 0, a_i \in A_i\}$ is the support of x_i. For each $x \in X$, player i's expected utility is $u_i(x) = \sum_{a \in A} u_i(a) \prod_j x_j(a_j)$ and $u_i(a_i, x_{-i}) = \sum_{a \in A} u_i(a) \prod_{j \in N \setminus \{i\}} x_j(a_j)$. Generally, $-i$ denotes the set of all players except player i.

2.1.1.1 Nash Equilibria

The Nash Equilibrium (NE) is an important solution concept for a game, which is a strategy profile x^* such that, for each player i, x_i^* is a best response to x_{-i}^* (i.e., $x_i^* = BR(x_{-i}^*)$ with $u_i(x_i^*, x_{-i}^*) \geq u_i(x_i, x_{-i}^*), \forall x_i \in X_i$). In zero-sum two-player games with $N = \{1, 2\}$ and $\sum_{i \in N} u_i(a) = 0 (\forall a \in A)$, an NE can be computed by solving the following linear program and its dual program:

$$\max_{x_1} U \tag{2.1a}$$

$$U \leq \sum_{a_1 \in A_1} u_T(a_1, a_2) x_1(a_1) \ \ \forall a_2 \in A_2 \tag{2.1b}$$

$$\sum_{a_1 \in A_1} x_1(a_1) = 1, x_1(a_1) \geq 0 \tag{2.1c}$$

For simplicity, we say that an NE is computed by solving Problem (2.1).

2.1.1.2 Team-Maxmin Equilibria

The Team-Maxmin Equilibrium (TME, and TMEs for the plural equilibria) [13, 14] is a solution concept for zero-sum multiplayer games with that a team $T = \{1, \ldots, n-1\}$ with $u_i(a) = u_j(a) (\forall i, j \in T, a \in A)$ and $\sum_{i \in T} u_i(a) = u_T(a) = -u_n(a) (\forall a \in A)$ plays against an adversary n, and each team player takes actions independently. We call G with such a scenario G_T. A TME is an NE with the properties that it is unique except for degenerate cases[1] and a best NE for the team. The utility of the team under the TME is called the TME value. In an ϵ-TME, the team and the adversary both cannot gain more than ϵ by the unilateral deviation of players, and the gap between the TME value and the ϵ-TME value is not greater than ϵ. The team-maxmin strategy profile x_T(i.e., $\times_{i \in T} x_i$)

[1]The situation of multiple TMEs can only occur in degenerate cases with special entries in the payoff matrix [13] because a TME gives the team the highest utility among all NEs.

in a TME (x_T, x_n) can be computed by the following nonlinear program:

$$\max_{x_1,\ldots,x_{n-1}} U \tag{2.2a}$$

$$U \leq \textstyle\sum_{a_T \in A_T} u_T(a_T, a_n) \prod_{i\in T} x_i(a_T(i)) \forall a_n \in A_n \tag{2.2b}$$

$$\textstyle\sum_{a_i \in A_i} x_i(a_i) = 1, x_i(a_i) \geq 0 \quad \forall i \in T \tag{2.2c}$$

where a_T is a joint action of the team (i.e., $\times_{i\in T} a_i$), $a_T(i)$ is the action of player i in a_T, and $A_T = \times_{i\in T} A_i$ is the set of these joint actions. After computing the team-maxmin strategy profile x_T, we can compte the adversary strategy x_n by minimizing the team's utility and making sure that no team members would like to deviate from their strategies in x_T [13]. Then x_n can be computed by solving the following linear program [13]:

$$\min_{x_n} \sum_{i\in T} z_i \tag{2.3a}$$

$$z_i - \sum_{a_n \in A_n} x_n(a_n) u_T(a_i, x_{T\setminus\{i\}}, a_n) \geq 0 \quad \forall i \in T, a_i \in A_i \tag{2.3b}$$

$$\sum_{a_n \in A_n} x_n(a_n) = 1 \tag{2.3c}$$

$$x_n(a_n) \geq 0 \quad \forall a_n \in A_n \tag{2.3d}$$

For simplicity, we say that a TME is computed by solving Problem (2.2).

The Correlated Team-Maxmin Equilibrium (CTME) [14] captures the situation where team players can correlate their actions, i.e., they can synchronize actions in G_T. That is, the team has the set of actions A_T, and the set of mixed strategies ($\overline{x}_T \in \Delta(A_T)$), and $\underline{A}_{i,\overline{x}_T} = \{a_i \mid a_i \in A_i, \exists a_T = (a_i, a_{T\setminus\{i\}}), \overline{x}_T(a_T) > 0\}$. The CTME maintains the properties of the NE in zero-sum two-player games, i.e., a CTME is equivalent to an NE in zero-sum two-player games where the team is teated as a single player. For example, CTMEs are exchangeable and a CTME can be computed by a linear program, i.e., the multilinear term $\prod_{i\in T} x_i(a_T(i))$ in Eq.(2.2b) is replaced by a single variable $\overline{x}_T(a_T)$:

$$\max_{\overline{x}_T} U \tag{2.4a}$$

$$U \leq \textstyle\sum_{a_T \in A_T} u_T(a_T, a_n)\overline{x}_T(a_T) \ \forall a_n \in A_n \tag{2.4b}$$

$$\textstyle\sum_{a_T \in A_T} \overline{x}_T(a_T) = 1, \overline{x}_T(a_T) \geq 0 \quad \forall i \in T \tag{2.4c}$$

In zero-sum multiplayer games with that a team $T = \{1, \ldots, n-1\}$ with $u_i(a) = u_j(a)(\forall i, j \in T, a \in A)$ and $\sum_{i \in T} u_i(a) = u_T(a) = -u_n(a)(\forall a \in A)$, let v_e be the utility value of the team under an NE, v_m be the utility value of the team under a TME, and v_c be the utility value of the team under a CTME. We have $v_e \leq v_m \leq v_c$ with that $\frac{v_m}{v_e}$ may be ∞, and $\frac{v_c}{v_m}$ may be ∞ [14].

2.1.2 Extensive-Form Games

An imperfect-information Extensive-Form Game (EFG) [4] is a tuple $(N, A, H, L, \chi, \rho, \mu, u, I)$, where: $N = \{1, \ldots, n\}$ is a set of players, A is a set of actions, H is a set of nonterminal nodes, L is a set of terminal nodes, $\chi : H \rightarrow 2^A$ is the action function assigning a set of possible actions to each nonterminal node, $\rho : H \rightarrow N$ is the player function assigning an acting player to each nonterminal node ($H_i = \{h \mid \rho(h) = i, h \in H\}, \forall i \in N$), $\mu : H \times A \leftarrow H \cup L$ is the successor function, which maps a nonterminal node and an action to a new node, $u = (u_1, \ldots, u_n)$ is the set of players' utility functions where $u_i : L \rightarrow \mathbb{R}$ assigns utilities to terminal nodes for player i, and $I = (I_1, \ldots, I_n)$ is the set of information sets where I_i is a partition of H_i such that, $\rho(h_1) = \rho(h_2)$ and $\chi(h_1) = \chi(h_2)$ for any $h_1, h_2 \in H_i$ whenever there exists $I_{i,j} \in I_i$ with $h_1 \in I_{i,j}$ and $h_2 \in I_{i,j}$. Without loss of generality, we assume, for each action $a \in A$, there is unique $I_{i,j}$ such that $a \in \chi(I_{i,j})$. In games with perfect recall, for each player i and each $I_{i,j} \in I_i$, nodes in $I_{i,j}$ share the same sequence of moves of player i on the paths from the root.

A behavioral strategy β_i defines a probability distribution over $\chi(I_{i,j})$ for each information set of player i. A sequence (σ_i) of actions of player i, defined by a node $h \in H \cup L$ of the game tree, is the ordered set of player i's actions that are on the path from the root to h. Let Σ_i define the set of sequences of player i. Let $\emptyset$ denote the fictitious sequence corresponding to the root. A realization plan $r_i : \Sigma \rightarrow [0, 1]$ is a function assigning a probability to be played to each sequence, which satisfies the following constraints.

$$r_i(\emptyset) = 1 \tag{2.5a}$$

$$\textstyle\sum_{a\in\chi(I_{i,j})} r_i(\sigma_i a) = r_i(\sigma_i) \;\; \forall I_{i,j} \in I_i, \sigma_i = \mathrm{seq}_i(I_{i,j}) \tag{2.5b}$$

$$r_i(\sigma_i) \geq 0 \quad \forall \sigma_i \in \Sigma_i \tag{2.5c}$$

where $\mathrm{seq}_i(I_{i,j})$ is the sequence leading to $I_{i,j}$. Let $\mathcal{R}_i$ be the set of all SFSs. We call r_i a pure SFS if $r_i(\sigma_i) \in \{0, 1\}(\sigma_i \in \Sigma_i)$, and $\overline{\mathcal{R}}_i$ the set of pure SFSs and $\overline{\mathcal{R}}_i \subseteq \mathcal{R}_i$.

Example 2.1. *In the 3-player Kuhn poker game, the information set includes the card the player has and the observed actions taking by players in turn. For example, J:/ccrc: is information set of player 2, where player 2 holds card J and she has observed the actions 'c,c,r,c' of all players in turn. Now, in information set J:/ccrc: reached by sequence J:/c:c of player* 2*, player* 2 *has two actions: calling (c) and folding (f). Therefore, by Eq.(2.5b), we have* r_2(*J:/c:c*) $=$ r_2(*J:/ccrc:c*) $+$ r_2(*J:/ccrc:f*).

A pure Normal-Form Strategy (NFS) of player i is a tuple $\pi_i \in \Pi_i = \times_{I_{i,j}\in I_i}\chi(I_{i,j})$ specifying an action to each information set of player i, and the size of Π_i is exponential in the size of the game tree. A pure reduced NFS is that it only specifies actions for reachable information sets due to earlier actions. Henceforth, we focus on reduced NFSs and just call them NFSs. A mixed NFS x_i is a probability distribution over Π_i, i.e., $x_i \in \Delta(\Pi_i)$. A pure NFS can be represented by a (unique) pure SFS, and a mixed NFS can be represented by an SFS (a convex combination of pure SFSs) [3]. Two strategies of player i are realization-equivalent (i.e., $\sim$) if both define the same probabilities for reaching nodes given any strategies of the other players, and they are realization-equivalent if and only if they have the same realization plan [3].

2.1.2.1 Nash Equilibria

In EFGs, an NE is a strategy profile r^* such that, for each player i, r_i^* is a best response to r_{-i}^* (i.e., $r_i^* = BR(r_{-i}^*)$ with $u_i(r_i^*, r_{-i}^*) \geq u_i(r_i, r_{-i}^*), \forall r_i \in \mathcal{R}_i$). In zero-sum two-player games with $N = \{1, 2\}$ and $\sum_{i\in N} u_i(\sigma) = 0(\forall a \in A)$, an NE can be computed by solving the following linear program:

$$\max_{r_1} v(\mathcal{I}_2(\emptyset)) \tag{2.6a}$$

$$v(\mathcal{I}_2(\sigma_2)) - \textstyle\sum_{I_{2,j} \in I_2 : \mathrm{seq}_2(I_{2,j}) = \sigma_n} v(I_{2,j})$$
$$\leq \textstyle\sum_{\sigma_1 \in \Sigma_1} u_1(\sigma_1, \sigma_2) r_1(\sigma_1) \quad \forall \sigma_1 \in \Sigma_1 \tag{2.6b}$$

$$\text{Eqs.}(2.5a) - (2.5c) \quad \forall i \in N \tag{2.6c}$$

2.1.2.2 Team-Maxmin Equilibria

In zero-sum multiplayer EFGs, a TME is solution concept for cases with $u_i(\sigma) = u_j(\sigma)(\forall i, j \in T, \sigma \in \Sigma)$ and $u_n(\sigma) = -u_T(\sigma) = -\sum_{i \in T} u_i(\sigma)(\forall \sigma \in \Sigma)$, where U_T denotes the team's utility with $U_T(\sigma) = \sum_{l \in L'} u_T(l)c(l)$ if at least a terminal node $l \in L' \subseteq L$ is reached by the joint plan σ (L' is the set of those terminal nodes) with the chance $c(l)$ determined by chance nodes, and $U_T(\sigma) = 0$ otherwise. A team-maxmin strategy profile in a TME in EFGs can be computed by the following non-convex program [15]:

$$\max_{r_1, \ldots, r_{n-1}} v(\mathcal{I}_n(\emptyset)) \tag{2.7a}$$

$$v(\mathcal{I}_n(\sigma_n)) - \textstyle\sum_{I_{n,j} \in I_n : \mathrm{seq}_n(I_{n,j}) = \sigma_n} v(I_{n,j})$$
$$\leq \textstyle\sum_{\sigma_T \in \Sigma_T} U_T(\sigma_T, \sigma_n) \prod_{i \in T} r_i(\sigma_T(i)) \quad \forall \sigma_n \in \Sigma_n \tag{2.7b}$$

$$\text{Eqs.}(2.5a) - (2.5c) \quad \forall i \in T \tag{2.7c}$$

where $\mathcal{I}_n(\sigma_n)$ is the information set where player n takes the last action of sequence σ_n, $v : I_n \to \mathbb{R}$ where $v(I_{n,j})$ is the team's expected utility in information set $I_{n,j}$, σ_T is the team's joint sequence (i.e., $\times_{i \in T} \sigma_i$), $\sigma_T(i)$ is the sequence of player i in σ_T, and Σ_T is the set of these joint sequences. By Eq.(2.6b), the adversary chooses the action minimizing the team's utility in each information set $\mathcal{I}_n(\sigma_n)$.

The Team-Maxmin Equilibrium with Coordination device (TMECor, and TMEsCor for the plural equilibria) [15, 28] is a solution concept capturing the scenarios where a single team $T = \{1, \ldots, n-1\}$ with *ex ante* coordination play against an adversary n with $u_i(l) = u_j(l)(\forall i, j \in T, l \in L)$ and $u_T(l) = \sum_{i \in T} u_i(l) = -u_n(l)(\forall l \in L)$ in a zero-sum EFG. Here, *ex ante* coordination means that team players can communicate only before the play through a coordination device. Then the team in a TMECor plays joint normal-form strategies. A team's mixed strategy $\overline{x}_T(\in \Delta(\Pi_T))$ is a probability distribution over $\times_{i\in T}\Pi_i = \Pi_T$ (the set of joint normal-form strategies and $\pi_T \in \Pi_T$). U_T denotes the utility of the team with:

$$U_T(\overline{x}_T, \sigma_n) = \textstyle\sum_{\pi_T\in\Pi_T} U_T(\pi_T, \sigma_n)\overline{x}_T(\pi_T) \tag{2.8}$$

where $U_T(\pi_T, \sigma_n) = \sum_{l\in L_{\pi_T,\sigma_n}} u_T(l)c(l)$, and $l(\in L_{\pi_T,\sigma_n} \subseteq L)$ is a terminal node reached by a strategy profile (π_T, σ_n) (L_{π_T,σ_n} is the set of those terminal nodes) with chance $c(l)$ due to chance nodes. Similarly,

$$U_T(\overline{x}_T, r_n) = \textstyle\sum_{\pi_T\in\Pi_T} U_T(\pi_T, r_n)\overline{x}_T(\pi_T) \tag{2.9}$$

where $U_T(\pi_T, r_n) = \sum_{l\in L_{\pi_T,r_n}} r_n(\text{seq}_n(l))u_T(l)c(l)$. A TMECor is equivalent to an NE in zero-sum two-player games where the team is teated as a single player, which can be computed by solving the following linear program:

$$\max_{\overline{x}_T} v(\mathcal{I}_n(\emptyset)) \tag{2.10a}$$

$$v(\mathcal{I}_n(\sigma_n)) - \textstyle\sum_{I_{n,j}\in I_n:\text{seq}_n(I_{n,j})=\sigma_n} v(I_{n,j}) \leq U_T(\overline{x}_T, \sigma_n) \quad \forall \sigma_n \in \Sigma_n \tag{2.10b}$$

$$\textstyle\sum_{\pi_T\in\Pi_T} \overline{x}_T(\pi_T) = 1 \tag{2.10c}$$

$$\overline{x}_T(\pi_T) \geq 0 \quad \forall \pi_T \in \Pi_T \tag{2.10d}$$

The Team-Maxmin Equilibrium with Communication device (TMECom, and TMEsCom for the plural equilibria) [15] is a solution concept capturing scenarios where a team of players with a communication device (similar to a mediator) and the same utility function play against an adversary. That is, all team players can communicate before and during the play through a communication device, which receives the information that the team members observed and sends recommendations related to the action to play at each information set for each team member. More specifically, team members can discuss and agree on tactics before the game starts, and they can communicate to

Algorithm 1: *ISG* for a two-player game (Vanilla-ISG)

```
1  Initialize $A'$.
2  repeat
3      $x \leftarrow CoreLP(A')$;
4      $a_1 \leftarrow$ Best response oracle of player 1 against $x_2$;
5      if $a_1 \notin A'_1$ then
6          $A'_1 \leftarrow A'_1 \cup \{a_1\}$
7      $a_2 \leftarrow$ Best response oracle of player 2 against $x_1$;
8      if $a_2 \notin A'_2$ then
9          $A'_2 \leftarrow A'_2 \cup \{a_2\}$
10 until convergence;
11 return $x$.
```

receive recommendations during the game. A TMECom is an NE in zero-sum multiplayer extensive-form games and has the properties of NEs in zero-sum two-player games (i.e., a TMECom is equivalent to an NE in zero-sum two-player games where the team is teated as a single player), which means that a TMECom can be computed by a linear program the same as the one for computing NEs in zero-sum two-player games (the team can be treated as a single player).

In zero-sum multiplayer EFGs with that $u_i(\sigma) = u_j(\sigma)(\forall i, j \in T, \sigma \in \Sigma)$ and $u_n(\sigma) = -u_T(\sigma) = -\sum_{i\in T} u_i(\sigma)(\forall \sigma \in \Sigma)$, let v_m be the utility value of the team under a TME, v_{cor} be the utility value of the team under a TMEcor, and v_{com} be the utility value of the team under a TMEcom. We have $v_m \leq v_{cor} \leq v_{com}$ with that $\frac{v_{com}}{v_m}$ may be ∞, $\frac{v_{com}}{v_{cor}}$ may be ∞, and $\frac{v_{cor}}{v_m}$ may be ∞ [15].

2.2 Scalable Approaches

In addition to the formulation to compute equilibria in the previous section, there are many scalable approaches to compute equilibria in two-player games efficiently. Here, we discuss two approaches that will be extended to multiplayer games in this thesis.

Algorithm 2: Iterative MDT

```
1 Choose $p = P = \lfloor \log_{10} y_j^U \rfloor$;
2 repeat
3     $\underline{f}_0 \leftarrow$ Solving Problem (2.12);
4     $\overline{f}_0 \leftarrow$ An upper bound of Problem (2.11), e.g., by using local non-linear
        program optimization algorithm to solve Problem (2.11);
5     if $\overline{f}_0 - \underline{f}_0 > \epsilon$ then
6         $p = p - 1$
7 until $\overline{f}_0 - \underline{f}_0 \leq \epsilon$;
8 return $x$.
```

2.2.1 Incremental Strategy Generation

The Incremental Strategy Generation (ISG) algorithm (including the double oracle algorithm, column generation, and so on) [5, 8, 20, 26, 38, 39] has shown the advantage for improving the scalability for computing an NE in two-player zero-sum games. The algorithm includes the following steps, repeating until convergence: 1) creating a restricted game G' by limiting the set of actions for each player i, i.e., $A'_i \subseteq A_i$; 2) computing the equilibrium strategy profile x^* in this restricted game G'; and 3) computing a best response a_i against x^*_{-i} in the original unrestricted game G for each player i, and add a_i to A'_i if $u_i(a_i, x^*_{-i}) > u_i(x^*)$. The algorithm terminates when no actions are added to the restricted game for all players (i.e., $u_i(a_i, x^*_{-i}) \leq u_i(x^*), \forall a_i \in A_i, i \in N$). Algorithm 1 shows an ISG algorithm for a two-player game.

2.2.2 An Iterative Algorithm for Bilinear Optimization

When a program involves bilinear terms, it is a non-convex program, which needs global optimization techniques to approximate it. It is well-known that the state-of-the-art global optimization solver is BARON [27]. Here we introduce a global optimization technique, the Multiparametric Disaggregation Technique (MDT) [40–42] for approximating bilinear terms by using a digit-wise discretization of one variable in the bilinear term and then transforming the non-convex program into a mixed-integer linear program (MILP), which is a strong competitor of BARON and has been applied to solve game-theoretical problems [43, 44] for two-player games. The MDT extends the McCormick relaxation [45] and outperforms the piecewise McCormick relaxation [40, 41]. More importantly, it has been applied for solving game-theoretical problems [43, 44].

Now we discuss the MDT shown in Kolodziej et al. (2013). A bilinear program is:

$$\min f_0 = \sum_{(i,j)\in BL_0} a_{i,j,0} y_i y_j + h_0(y) \tag{2.11a}$$

$$f_1 = \sum_{(i,j)\in BL_q} a_{i,j,q} y_i y_j + h_q(y) \quad q \in Q \setminus \{0\} \tag{2.11b}$$

$$y \in S \cap \Omega \in \mathbb{R}^n \tag{2.11c}$$

where h_q is linear, $a_{i,j,q}$ is a scalar with $i \in I, j \in J$, and $q \in Q = \{0, 1, \ldots, n\}$ representing the set of all functions f_q. BL_q is a set of bilinear terms $y_i y_j$ in function f_q with $i \neq j$. $S \in \mathbb{R}^n$ is convex and $\Omega \in \mathbb{R}^n$ is an n-dimensional hyperrectangle with bounds, i.e., $\Omega = \{y \in \mathbb{R}^n : 0 \leq y^L \leq y \leq y^U\}$. By using the MDT, we can transform the above non-convex program into the following MILP:

$$\min f_0 = \sum_{(i,j)\in BL_0} a_{i,j,0} w_{i,j} + h_0(y) \tag{2.12a}$$

$$f_1 = \sum_{(i,j)\in BL_q} a_{i,j,q} w_{i,j} + h_q(y) \quad q \in Q \setminus \{0\} \tag{2.12b}$$

$$w_{i,j} = \sum_{l=p}^{P} \sum_{k=0}^{9} 10^l \cdot k \cdot \hat{y}_{i,j,k,l} + \Delta w_{i,j} \quad \forall (i,j) \in BL_q, q \in Q \tag{2.12c}$$

$$y_j = \sum_{l=p}^{P} \sum_{k=0}^{9} 10^l \cdot k \cdot z_{j,k,l} + \Delta y_j \quad \forall j \in \{j \mid (i,j) \in BL_q, q \in Q\} \tag{2.12d}$$

$$y_i = \sum_{k=0}^{9} \hat{y}_{i,j,k,l} \quad \forall (i,j) \in BL_q, q \in Q, l \in L \tag{2.12e}$$

$$y_i^L \cdot z_{j,k,l} \leq \hat{y}_{i,j,k,l} \leq y_i^U \cdot z_{j,k,l} \quad \forall (i,j) \in BL_q, q \in Q, l \in L, k \in K \tag{2.12f}$$

$$\sum_{k=0}^{9} z_{j,k,l} = 1 \quad \forall j \in \{j \mid (i,j) \in BL_q, q \in Q\}, l \in L \tag{2.12g}$$

$$y_i^L \cdot \Delta y_j \leq \Delta w_{i,j} \leq y_i^U \cdot y_j \quad \forall (i,j) \in BL_q, q \in Q \tag{2.12h}$$

$$\Delta w_{i,j} \leq (y_i - y_i^L) \cdot 10^p + y_i^L \cdot \Delta y_j \quad \forall (i,j) \in BL_q, q \in Q \tag{2.12i}$$

$$\Delta w_{i,j} \geq (y_i - y_i^U) \cdot 10^p + y_i^U \cdot \Delta y_j \quad \forall (i,j) \in BL_q, q \in Q \tag{2.12j}$$

$$0 \leq \Delta y_j \leq 10^p \quad \forall j \in \{j \mid (i,j) \in BL_q, q \in Q\} \tag{2.12k}$$

$$z_{j,k,l} \in \{0,1\} \quad \forall j \in \{j \mid (i,j) \in BL_q, q \in Q\}, k \in K, l \in L \tag{2.12l}$$

$$y \in S \cap \Omega \in \mathbb{R}^n \tag{2.12m}$$

where $L = \{p, p+1, \ldots, P\}$ is a set of integers, $K = \{0, \ldots, 9\}$ is a base of 10 for discretizing y_j (Eqs.(2.12d) and (2.12g)), Δy_i is a slack variable of y_i bounded between 0 and 10^p (Eq.(2.12k)), $\hat{y}_{i,j,k,l} = y_i \cdot z_{j,k,l}$ (Eq.(2.12f)) with the corresponding representation of y_i in Eq.(2.12e), $w_{i,j} = y_i y_j$ with its slack variable $\Delta w_{i,j} = y_i \cdot \Delta y_j$ represented by the McCormick relaxation in Eqs.(2.12h)–(2.12j). Due to the slack variable of $w_{i,j}$, the optimal objective value of this MILP is a lower bound of the original bilinear program.

An iterative MDT algorithm is shown in Algorithm 2, which starts from some coarse precision level, and then iteratively increases the precision level when the difference between the lower and upper bounds is not small enough.

2.3 Related Work

In this section, we review existing research relevant to this thesis to highlight the novelty of our work.

This thesis computes TMEs and the corresponding variants, where more specific restrictions or information is available. The TME is introduced by von Stengel and Koller (1997), where computing a TME in normal-form games is formulated as a nonlinear program. Hansen et al. (2008) show that computing a TME is FNP-hard. Basilico et al. (2017) study the relation between the TME and the CTME in terms of efficiency. They also propose a heuristic algorithm (we will discuss it when we compute TMEs) to compute a TME through transforming the strategy of the team in a CTME, which may cause a large loss for the team. Celli and Gatti (2018) study the TME with its variants (the TMECor and the TMECom) in extensive-form games and the relation between the TME and its variants in terms of efficiency and show that computing a TME or a TMECor in extensive-form games is FNP-hard. Celli and Gatti (2018) propose a column generation algorithm (an ISG algorithm) to compute a TMECor, and Farina et al. (2018) propose a learning algorithm based on the fictitious play framework to compute a TMECor (we will discuss them when we compute TMEsCor), which are still inefficient to solve large games.

2.3.1 Converging to Team-Maxmin Equilibria

McMahan *et al.* [5] propose the first ISG (also called the double oracle algorithm) for zero-sum two-player games, where the robot chooses a path to a goal location while avoiding being detected by an adversary on the road. Given the adversary strategy, the robot's best response oracle (to compute a best response against the adversary strategy) is modelled as a Markov Decision Process (MDP). Then solving the best response oracle is equivalent to solving the corresponding MDP. After that, ISG is used to solve many similar problems, including the classic network security games [20, 21, 23] and extensive-form games [38]. The MDP feature of the best response oracle in ISG makes it possible to deploy deep reinforcement learning [46] and be extended to multiagent learning [47, 48]. In this paper, we study the problem of extending ISG for converging to TMEs, which will be a base for developing learning algorithms for TMEs in multiplayer games.

However, the extension is not straightforward. It is well-known that ISG converges to an NE in zero-sum two-player games [5], but, which is unclear (to our best knowledge) in multiplayer games. Then we theoretically show that ISG (Vanilla-ISG) converges to an NE in multiplayer games. However, as we will illustrate, the existing ISG cannot guarantee to converge to a TME. Then we try to extend it to ISGT for converging to a TME. As we will illustrate, it is difficult to converge to a TME, e.g., we cannot simply add the team's best response to the restricted game to converge to a TME. Then we add our new operations to ISG to guarantee to converge to a TME by exploiting that a TME is an NE maximizing the team's utility. We also show that the conditions in our operations cannot be further relaxed. Finally, we theoretically show that our new ISG (i.e., ISGT) can guarantee to converge to a TME.

The Correlated TME (CTME) [14] is a solution concept close to the TME, where team players with the same utility function can synchronize their actions against the adversary. That is, team players can jointly plan and execute their strategies, which means that the team is equivalent to a single player with actions as the joint team action profiles. Then, a CTME can be found through a linear program similar to finding an NE in zero-sum two-player games. However, team players in a TME cannot correlate their actions [15], and then cannot directly use the strategies in a CTME. To compute a TME, one approach is that the team can compute a CTME first and then transform the team's correlated strategy into the team's mixed strategy profile, where a transformation algorithm was proposed [14]. However, this transformation cannot theoretically guarantee to obtain a team-maxmin strategy profile (in a TME) for the team and may cause a huge loss for the team, as shown in the previous experiments [14]. We study the limitations of computing TMEs based on CTMEs. We first theoretically show that using the strategy transformed from a CTEM may cause an arbitrarily large loss to the team. Second, we show that a TME in a restricted game with the strategies for computing a CTME may not be an NE in the full game, and it may not be a TME in the full game even it is an NE in the full game. These results show that it is not straightforward to compute a TME via computing a CTME. Therefore, we develop our novel operations to improve the scalability of our algorithm ISGT by computing a CTME to initialize the strategy space and exploiting the team's utility in a CTME to terminate earlier. Finally, we theoretically show that our new CTME based ISG (e.g., CISGT) can guarantee to converge to a TME.

2.3.2 Optimal Escape Interdiction on Transportation Networks

This type of problems is typically referred to as pursuit-evasion games (PEGs) within the game theory literature, where the evader tries to minimize the probability of encountering the pursuer but the pursuer wants to thwart the evader's plans by capturing him [49, 50]. However, PEGs do not consider realistic traffic dynamics, where the travel time is influenced by the traffic flow. Furthermore, the evader in PEGs cannot escape to the external world, thus the goal of the pursuer is to minimize the number of rounds needed to catch the evader. In contrast, our domain consists of a defender who aims to maximize the probability of capturing the attacker before he escapes. While there has been a variety of research work on applying game theoretic approaches to security domains [39, 51–58], most of these unrealistically assume that at most one player takes paths [59, 60], and the time dynamics are irrelevant [25, 61]. As such, these models are not suitable for our problem. In particular, the attacker can exploit the fact that traffic is slower in some places, and then can choose a path, which could have been covered by the police in current models with no time dynamics, but not in our setting.

2.3.3 Computing Team-Maxmin Equilibria in Extensive-Form Games

The solution concept similar to the TME is the CTME, where team players correlate their strategies by exploiting a mediator recommending actions to them through communication [14]. CTMEs in EFGs [15] include: 1) TMEs with Communication device (TMECom) for the situation where team players can communicate and correlate their actions before the play of the game and during the game's execution; and 2) TMEs with Coordination device (TMECor) for the situation where team players communicate and correlate their actions only before the play (ex ante coordination [28]). A TMECom be computed in polynomial time, but computing a TMECor is FNP-hard, where each team member's (normal-form) strategy space is exponential in the size of EFGs and efficient algorithms [15, 28] are proposed. However, team players in a TME have no communication and then do not correlate their actions [15]. To compute a TME in E-FGs, one approach is to compute a CTME first and then transform it into the mixed strategy profile of the team, motivated by the algorithm in normal-form games [14]. Unfortunately, this approach may give the team a huge loss because of the large gap between the team's utility obtained by this approach and the TME value as shown in the previous experiments [14]. We theoretically show that this loss can be arbitrarily large.

In the domain of global optimization, the Multiparametric Disaggregation Technique (MDT) [40–42] for approximating bilinear terms is a strong competitor of BARON. The MDT outperforms the piecewise McCormick relaxation [40, 41] and has been applied to solve game-theoretical problems [43, 44]. To approximate multilinear terms, we can recursively transform multilinear terms to bilinear terms and then apply the MDT inspired by the recursive McCormick relaxation [62]. However, the current MDT approach cannot compute TMEs efficiently because the number of bilinear terms in EFGs is large (e.g., there are hundreds of bilinear terms in small Kuhn poker games) and the MDT will use a large number of constraints and integer variables for approximating each term. Therefore, instead of applying the MDT directly, we develop novel techniques.

2.3.4 Computing Team-Maxmin Equilibria with Coordination Device

The Team-Maxmin Equilibrium (TME) [13, 15] is a solution concept close to the TMECor, where a team of players with the same utility function independently play against an adversary. The TME in extensive-form games assumes each team member takes behavioral strategies, which comes at a high cost compared to the TMECor [15, 28]. Firstly, computing a TME, i.e., finding the optimal joint behavioral strategies of team members, is FNP-hard and a non-convex optimization problem [15]. Secondly, the team may lose a large utility if they use the TME instead of the TMECor [15, 28]. Moreover, the team as a whole has imperfect recall due to the lack of communication between team members during the game, where behavioral strategies cannot capture the correlation between the normal-form strategies of team members [28]. That is, generally, behavioral strategies are not realization-equivalent to normal-form strategies induced by the coordination device, and then can cause a large loss of utilities to the team [28]. In fact, we cannot even guarantee the existence of NEs in these behavioral strategies [63]. Therefore, using normal-formal strategies is crucial to a TMECor.

A TMECor can be computed via normal-form strategies for all players, but each player's normal-form strategy space is exponential in the size of the game tree. To compute TMEsCor efficiently, Celli and Gatti [15] propose a hybrid representation, where the team plays joint normal-form strategies while the adversary plays sequence-form strategies. Farina *et al.* [28] develop a realization-form strategy representation for all players focusing on the probability distribution on terminal nodes of the game tree, and then use it to derive an auxiliary game representing the original game through a set of subtrees. Each subtree is a two-player game between one team member and the adversary by playing realization-form strategy[1], where the remaining team members fix a joint pure normal-form strategy. Although the auxiliary game has a hybrid-form structure for the team, it is not a strategy representation and, most importantly, is based on the realization-form strategy that is not able to be executed from the root of the game tree. To make a realization-form strategy executable, they need to reconstruct a normal-form strategy from it. Unfortunately, it is not easy to do that in large games. For example, an only known existing reconstruction algorithm (Algorithm 2 in Celli *et al.* [29]) runs in time $O(|L|^2)$ (L is the set of terminal nodes in a game), which is extremely large in large games (see Table 6.1). To avoid such a cumbersome reconstruction algorithm, following the above hybrid form, we propose our new strategy representation for the team, which is able to be executed from the root immediately. That is, in our hybrid-form strategies, one team member takes sequence-form strategies (instead of realization-form strategies) while other team members take pure normal-form strategies. However, it is unclear whether our strategy representation with an infinite number of pure strategies (due to the sequence-form strategies of one member) can be used to compute exact TMEsCor.[2] Fortunately, we theoretically show that our Column Generation (CG) algorithm can guarantee to converge to a TMECor within a finite number of iterations in our infinite strategy space. Therefore, although the similar structure of our hybrid-form strategy has been mentioned in the literature, we are the first to make it become the team's strategy representation without depending on the inexecutable realization-form strategies and study the theoretical property of using it to compute exact TMEsCor.

BR is an important component in CG to compute TMEsCor. The first proposed BR for computing TMEsCor [15] focuses on whether or not a leaf is reached by a

[1]Although this team member's realization-form strategy can be induced by a behavioral strategy in the subtree, their theoretical result for the equivalence between the auxiliary game and the original game is based on the realization-form strategy, and their algorithm Fictitious Team-Play (FTP) [28] also assumes that this team member takes a realization-form strategy and then still needs to transform it into a normal-form strategy to execute it from the root. Note that, although this team member's strategy in the output of the MILP representing BR of FTP is a sequence-form strategy, which is still transformed into a realization-form strategy in FTP.

[2]They [28] do not consider this issue for the continuous realization-form strategy space because they aim to compute approximate TMEsCor.

	New representation	BR		
		Integer variables	Compatible	Reduce space
Celli and Gatti [15]	Yes	$\|L\|$	Yes	No
Farina *et al.* [28]	Yes	$\sum_{i\in T\setminus\{1\}} \|\Sigma_i\|$	No	No
Ours	Yes	$\sum_{i\in T\setminus\{1\}} \|\Sigma_i\|$	Yes	Yes

TABLE 2.1: Comparing with existing approaches to compute TMEsCor in terms of the new strategy representation, the number of integer variables ($|L|$ is significantly larger than $\sum_{i\in T\setminus\{1\}} |\Sigma_i|$ in extensive-form games) in BR, the compatibility of BR with associated constraints, and reducing the feasible solution space of BR.

pure joint normal-form strategy of all team members, which has $|L|$ integer variables and then can only solve small games. Farina *et al.* [28] develop a faster BR focusing on the utility of the joint pure normal-form strategy of all team members except one given the realization-form strategy of other players. However, although the number of integer variables is significantly reduced to the number of sequences of all team members except one, this BR is still inefficient in large games with a large number of sequences. To speed up, one possible approach is to add associated constraints [16] to reduce the feasible solution space of the MILP representing BR.[1] Unfortunately, their BR [28] is not compatible with associated constraints. Therefore, we cannot directly add associated constraints to this BR. Overall, the main difference between different BR approaches is about how to represent the team's reaching probabilities for terminal nodes with utilities, which affects the number of integer variables and the compatibility with associated constraints. Our formulation is the first BR that can reduce the number of integer variables and can be compatible with associated constraints () at the same time (see Table 2.1).

Associated constraints are used to reduce the feasible solution space of an MILP for computing a TME in Zhang *et al.* [16]. These associated constraints are generated through a global optimization technique, which is based on the recursive McCormick relaxation [45, 62]. In our case, we can use the recursive McCormick relaxation to exactly transform the multilinear BR into an MILP. However, we then need to recursively generate associated constraints and then generate associated constraints from the terminal nodes to the root, which needs to generate new terms for sequences reaching nonterminal nodes frequently, similar to the algorithm in Zhang *et al.* [16]. When the

[1] Adding associated constraints, to our best knowledge, is the only known effective method to do that.

number of team players is larger, the number of variables in each multilinear term is larger, which needs much more recursive operations. This recursive generation of associated constraints undermines the effort to improve the speed by reducing the feasible solution space when the number of team players is large. To conquer this challenge, we develop our global optimization technique ART without depending on the recursive method, which efficiently generates associated constraints.

2.3.5 Optimal Interdiction of Urban Criminals with the Aid of Real-Time Information

The pursuit-evasion game (PEG) typically assumes that the evader knows the locations of the pursuer's units, but the pursuer does not know the location of the evader [35, 64]. While our model is a variant of PEG, we focus on a realistic setting in our motivating scenario where the defender has the adversary's real-time location information. Most of the work on PEGs cannot capture the special structure of specific problems. For example, although the partially observable stochastic game [65] is domain-independent, their model focuses on infinite-horizon games. Similarly, the patrolling security game (PSG) [36, 66], where the defender defends an environment against an unseen intruder who needs multiple turns to perform the attack, is typically modeled as a stochastic game with infinite horizon. Recently, PSGs have been extended to cover the situation that the defender receives a spatially uncertain signal after being attacked [67, 68], and has to reach the attacked target to catch the attacker. However, this signal response game does not consider real-time information after receiving the signal due to the fact that no more information is available for the defender after receiving the signal.

ISG has been successfully used to exactly solve many games [20, 34, 37, 38], whose core part is to compute the best response strategy. However, the existing exact solution approaches for both imperfect recall games and perfect recall games are unable to solve NEST efficiently. Firstly, NEST is similar to an imperfect recall game NFGSS [37], but NFGSS does not consider real-time information. Note that the general form of imperfect recall games cannot be exactly solved by a linear program [44]. Although NFGSS is solved by combining *ISG* with the network-flow representation of strategies

[69], its algorithm is impractical for the large NEST as their *ISG*'s *BR* with a depth-first search does not set bounds to cut branches and then evaluates all related states to compute the best response. Secondly, if NEST is modeled as a perfect recall game [3], it will lead to a significantly larger strategy space than the imperfect recall approach (see the discussion in [37]). Moreover, the branch-and-bound *BR* in the latest exact algorithm for perfect recall games [38] is still impractical for the large NEST because its bounds are very loose. More specifically, in their *BR*, the upper bound for each node always equals the maximum possible value of the game, and the lower bound for each succeeding node of the first node in the information set for the searching player always equals the minimum possible value of the game. The immediate result is that at least one succeeding state of each action will be evaluated. However, it is impractical to evaluate at least one succeeding state for each action in each state in NEST because NEST has an extremely large state space and a large action space for each state in the MDP for the defender.

Chapter 3

Converging to Team-Maxmin Equilibria in Zero-Sum Multiplayer Games

This chapter[1] focuses on computing TMEs [13–16] in normal-form games. Due to the hardness to solve the nonlinear program to find a TME, we develop our ISG algorithm to compute a TME. However, the study of ISG for computing TMEs is completely unexplored. In this chapter, we first study the properties of ISG for multiplayer games, showing that ISG converges to an NE but may not converge to a TME, and it can cause an arbitrarily large loss to the team. Second, we design an ISG variant (ISGT) by exploiting that a TME is an NE maximizing the team's utility and show that ISGT converges to a TME and the impossibility of relaxing conditions in ISGT. Third, to further improve the scalability, we design an ISGT variant (CISGT) by using the strategy space for computing an equilibrium that is close to TME but is easier to be computed as the initial strategy space of ISGT. Finally, extensive experimental results show that CISGT is orders of magnitude faster than ISGT and the state-of-the-art algorithm to compute TMEs in large games.

[1]The work in this chapter has been published as **Youzhi Zhang** and Bo An. Converging to team-maxmin equilibria in zero-sum multiplayer games. *Proceedings of the 37th International Conference on Machine Learning* (**ICML**), accepted, 2020.

3.1 ISG in Multiplayer Games

In this section, we show that Vanilla-ISG converges to an NE in multiplayer games. However, we show that Vanilla-ISG cannot guarantee to converge to a TME. Then, we provide a method (ISGT) to amend it by exploiting the property that a TME is an NE maximizing the team's utility.

In this chapter, G is called G_T if the TME can be deployed. Without loss of generality, we assume that G'_T with $A' = A'_T \times A'_n$ is a restricted game of G_T with $A'_i \subseteq A_i (\forall i \in N)$.

3.1.1 Limitations of Vanilla-ISG

We first show that Vanilla-ISG can converge to an NE but cannot guarantee to converge to a TME (even cannot guarantee to approximate a TME).

Theorem 3.1. *Vanilla-ISG converges to an NE in G.*

Proof. First, Vanilla-ISG will converge because the action set for each player is finite in G. Second, when Vanilla-ISG converges to strategy profile x^*, $u_i(a_i, x^*_{-i}) \leq u_i(x^*)$ ($\forall a_i \in A_i, i \in N$), which means that $u_i(x^*_i, x^*_{-i}) \geq u_i(x_i, x^*_{-i})$ ($\forall x_i \in X_i, i \in N$). Therefore, x^* is an NE. □

In multiplayer games, there are many different NEs, which are not exchangeable and give different utilities to players. However, a TME is an NE giving the best utility for the team, which is not considered by Vanilla-ISG. Indeed, Vanilla-ISG may not converge to a TME.

Proposition 3.1. *Vanilla-ISG may not converge to a TME in G_T.*

Proof. By Theorem 3.1, Vanilla-ISG converges to an NE in G_T. Now we only need to show that this NE may not be a best NE for the team. Consider G_T with three players (two teammates), two actions ({1,2}) per player, and the following utility function:

$$u_T(a) = \begin{cases} 10 & \text{if } a = (1,2,2) \text{ or } (2,1,1) \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

Suppose that the restricted game G'_T in Vanilla-ISG is initialized with action sets $A'_1 = \{1\}, A'_2 = \{2\}, A'_3 = \{2\}$. In G'_T, the single equilibrium is a pure strategy profile $(1, 2, 2)$. Then, A'_3 is expanded to $\{1, 2\}$ because $u_3(1, 2, 1) = 0 > -10 = u_3(1, 2, 2)$, while A'_1 and A'_2 are not expanded. Now the pure strategy profile $(1, 2, 1)$ is the new equilibrium in the new G'_T with $u_T(1, 2, 1) = 0$. For each player, A'_i will not be expanded because players 1 and 2 will gain 0 from the unilateral deviation, while player 3 will lose utility 10 from the unilateral deviation. That is, pure strategy profile $(1, 2, 1)$ is an NE in G_T, i.e., Vanilla-ISG converges to an NE.

Consider the mixed strategy profile $x = (x_1, x_2, x_3)$ with $x_i = (0.5, 0.5)$, which is an NE with utility 2.5 for the team because each player is indifferent between playing their actions with their utility under x, e.g., $u_T(1, x_{-1}) = u_T(2, x_{-1}) = 2.5$. Therefore, pure strategy profile $(1, 2, 1)$ is not a best NE for the team, which means that pure strategy profile $(1, 2, 1)$ is not a TME, concluding the proof. $\square$

Vanilla-ISG cannot guarantee to converge to a TME and also may not approximate a TME by the following result.

Proposition 3.2. *Vanilla-ISG can cause an arbitrarily large loss to the team in G_T.*

Proof. Consider G_T with utilities shown in Eq.(3.1). As shown in the proof for Theorem 3.1, Vanilla-ISG can converge to the NE $(1, 2, 1)$ (pure strategy profile) with utility 0 for the team while $x = (x_1, x_2, x_3)$ with $x_i = (0.5, 0.5)$ is another NE with utility 2.5 for the team. Therefore, Vanilla-ISG can cause an arbitrarily large loss to the team because $\frac{2.5}{0} = \infty$. $\square$

3.1.2 The Difficulty of Converging to a TME

Vanilla-ISG cannot guarantee to converge to a TME, so we need to extend the current ISG to converge to a TME. This section shows the difficulty of converging to a TME.

Vanilla-ISG's failure to converge to a TME shows that we cannot simply add each player's best response to obtain a TME. One straightforward extension of this idea is that we add the team's best response to the restricted game. However, the following result shows that this extension cannot guarantee to converge to a TME.

Proposition 3.3. *x may not be a TME in G_T if x is a TME in G'_T and an NE in G_T, and $\nexists a_T \in (A_T \setminus A'_T)$ such that $u_T(a_T, x_n) > u_T(x)$.*

Proof. Consider G_T with three players (two teammates), $A_2 = \{1, 2, 3\}$, $A_1 = A_3 = \{1, 2\}$, and the following utility function:

$$u_T(a) = \begin{cases} 10 & \text{if } a = (1,1,1) \text{ or } (2,2,2) \\ 5 & \text{if } a = (2,3,1) \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

In G'_T, $A'_1 = A'_2 = A'_3 = \{1, 2\}$. Let player i's mixed strategy be $x_i = (x_i(1), x_i(2))$. Given x_1, x_2, and the adversary's action 1, the team's utility is:

$$u_T(x_1, x_2, 1) = 10x_1(1)x_2(1)$$

Given x_1, x_2, and the adversary's action 2, the team's utility is:

$$u_T(x_1, x_2, 2) = 10(1 - x_1(1))(1 - x_2(1)) = 10(1 - x_1(1) - x_2(1) + x_1(1)x_2(1))$$

To achieve the TME value, we need to maximize the minimum of $u_T(x_1, x_2, 1)$ and $u_T(x_1, x_2, 2)$ (see Eq.(2.2)). The case that $u_T(x_1, x_2, 1) = u_T(x_1, x_2, 2)$ gives the largest minimum to the team. Let $u_T(x_1, x_2, 1) = u_T(x_1, x_2, 2)$, we have $x_1(1) = 1 - x_2(1)$. Then we have:

$$u_T(x_1, x_2, 1) = 10x_1(1)x_2(1) = 10(-(x_2(1) - 0.5)^2 + 0.25)$$

which has its maximum 2.5 at $(x_2(1) = 0.5$. Then we have $(x_1(1) = 0.5$. Now we have the team-maxmin strategy profile $x_T = (x_1, x_2) = ((0.5, 0.5), (0.5, 0.5))$ with the team utility 2.5 in G'_T. Given this x_T, we can achieve $x_3 = (0.5, 0.5)$ by the following equation (see Eq.(2.3)):

$$u_T(x_1, 1, x_3) = u_T(x_1, 2, x_3) \Rightarrow 0.5 \times 10 \times x_3(1) = 0.5 \times 10 \times (1 - x_3(1))$$

Then $x = (x_1, x_2, x_3)$ is a TME in G'_T. Note that $u_T(x_1, 3, x_3) = 0.5 \times 0.5 \times 5 = 1.25 < 0.5$. Therefore, strategy profile $x = (x_1, (0.5, 0.5, 0), x_3)$ is an NE in G_T. In addition, we have $u_T(2, 3, x_3) = 2.5$, which is not larger than $u_T(x) = 2.5$. However, strategy profile $x' = ((0, 1), (0, \frac{1}{3}, \frac{2}{3}), (\frac{2}{3}, \frac{1}{3}))$ is an NE in G_T with utility $\frac{10}{3}(> 2.5)$ for the team. The reason is that $u_T(1, x'_2, x'_3) = 0 < \frac{10}{3}$, and $u_T(x'_1, 1, x'_3) = 0 < \frac{10}{3} = u_T(x'_1, 2, x'_3) = u_T(x'_1, 3, x'_3) = u_T(x'_1, x'_2, 1) = u_T(x'_1, x'_2, 2)$. Therefore, x is not a TME in G_T. □

The reason for the above failure is that the adversary tries to avoid the strategy that gives a high utility for the team, which gives a low utility for himself in zero-sum games. Then we need to add more actions to the restricted game, instead of only adding the team's best response (i.e., adding a_T such that $a_T = \arg\max_{a'_T \in A_T} u_T(a_T, x_n)$ and $u_T(a_T, x_n) > u_T(x)$, where x is a TME in G'_T) or all of the team's better responses (i.e., adding a_T such that $u_T(a_T, x_n) > u_T(x)$). To do that, we can add the team's joint actions with outcomes that are better than the utility of the equilibrium in the restricted game (i.e., adding a_T such that $u_T(a_T, a_n) > u_T(x)$). However, there may be too many joint actions satisfying this condition in the full game, and adding too many actions to the restricted games will make it hard to compute a TME. In ISG, we compute the team's best response against the adversary's strategy and add it (only one joint action) to the restricted game at each iteration. To speed up, two straightforward extensions of this idea are that: 1) we only add joint actions related to the support set of the adversary's strategy, i.e., adding $(a_T, a_n) \in (A_T \setminus A'_T) \times \underline{A}_{n,x_n}$ such that $u_T(a_T, a_n) > u_T(x)$ to G'_T; and 2) we only add one joint action that can affect the TME value at each iteration. The following results show that both extensions cannot guarantee to converge to a TME.

Proposition 3.4. *x may not be a TME in G_T if x is a TME in G'_T and an NE in G_T, and $\nexists (a_T, a_n) \in (A_T \setminus A'_T) \times \underline{A}_{n,x_n}$ such that $u_T(a_T, a_n) > u_T(x)$.*

Proof. Consider G_T with three players (two teammates), $A_1 = A_2 = \{1,2,3\}$, $A_3 = \{1,2\}$, and the following utility function:

$$u_T(a) = \begin{cases} 3 & \text{if } a = (1,1,1) \text{ or } (1,1,2) \\ 21 & \text{if } a = (2,2,1) \\ 3 & \text{if } a = (2,3,1), (3,2,1), \text{ or } (3,3,1) \\ 10 & \text{if } a = (2,3,2), (3,2,2), \text{ or } (3,3,2) \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

In G'_T, $A'_1 = A'_2 = A'_3 = \{1,2\}$. Obviously, pure strategy profile $a = (1,1,1)$ is an NE in G'_T and G_T. Now we check that x is a TME in G'_T. Let player i's mixed strategy be $x_i = (x_i(1), x_i(2))$. Given x_1, x_2, and the adversary's action 1, the team's utility is:

$$u_T(x_1, x_2, 1) = 3x_1(1)x_2(1) + 21(1 - x_1(1))(1 - x_2(1))$$

Given x_1, x_2, and the adversary's action 2, the team's utility is:

$$u_T(x_1, x_2, 2) = 3x_1(1)x_2(1)$$

We can see that, if $x_1(1) < 1$ and $x_2(1) < 1$, action 1 is strictly dominated by action 2 for the adversary. However, given the adversary action 2 in G'_T, the team cannot achieve the utility that is higher than $u_T(a) = 3$. Now, given any strategy of the adversary, if player 1's strategy is $x_1(1) = 1$, player 2's best response is $x_1(1) = 1$, and vice versa. In this case, $a = (1,1,1)$ (i.e., $x = ((1,0),(1,0),(1,0))$) is a TME in G'_T. Given x, $\nexists(a_T, a_n) \in (A_T \setminus A'_T) \times \underline{A}_{n,x_n}$ such that $u_T(a_T, a_n) > u_T(x)$. However, strategy profile $x' = ((0,0.5,0.5),(0,0.5,0.5),(\frac{5}{14},\frac{9}{14}))$ is an NE in G_T with utility $7.5 (> 3)$ for the team. The reason is that $u_T(1, x'_2, x'_3) = 0 < 7.5 = u_T(2, x'_2, x'_3) = u_T(3, x'_2, x'_3)$, and $u_T(x'_1, 1, x'_3) = 0 < 7.5 = u_T(x'_1, 2, x'_3) = u_T(x'_1, 3, x'_3) = u_T(x'_1, x'_2, 1) = u_T(x'_1, x'_2, 2)$. Therefore, x is not a TME in G_T. □

Proposition 3.5. *x may not be a TME in G_T if x is a TME in G'_T and an NE in G_T, and $\nexists(a_T, a_n) \in (A_T \setminus A'_T) \times A'_n$ with $u_T(a_T, a_n) > u_T(x)$ such that the TME value in G'_T changes after adding a_T to A'_T.*

Proof. Consider G_T with three players (two teammates), $A_1 = A_2 = \{1, 2, 3\}$, $A_3 = \{1, 2\}$, and the following utility function:

$$u_T(a) = \begin{cases} 3 & \text{if } a = (1,1,1) \text{ or } (1,1,2) \\ 100 & \text{if } a = (2,2,1) \text{ or } (3,3,2) \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

In G'_T, $A'_1 = A'_2 = \{1\}, A'_3 = \{1, 2\}$. Obviously, $x = ((1), (1), (0.5, 0.5))$ is an NE in G'_T and G_T with utility 3 for the team, which is also a TME in G'_T. If we add $a_T = (2, 2)$ or $(3, 3)$, according to the analysis on the case in Eq.(3.3), x with $x_1(1) = 1 = x_2(1)$ and $x_3(1) = x_3(2) = 0.5$ is still a TME in G'_T with the TME value 3. That is, the TME value in G'_T does not change. However, strategy profile $x' = ((0, 0.5, 0.5), (0, 0.5, 0.5), (0.5, 0.5)$ is an NE in G_T with utility $25 (> 3)$ for the team. The reason is that $u_T(1, x'_2, x'_3) = 0 < 25 = u_T(2, x'_2, x'_3) = u_T(3, x'_2, x'_3)$, and $u_T(x'_1, 1, x'_3) = 0 < 25 = u_T(x'_1, 2, x'_3) = u_T(x'_1, 3, x'_3) = u_T(x'_1, x'_2, 1) = u_T(x'_1, x'_2, 2)$. Therefore, x is not a TME in G_T. □

3.1.3 Converging to a TME

Based on the discussion in the previous section, this section proposes our ISG algorithm for converging to a TME (ISGT). In fact, it exploits that a TME is an NE maximizing the team's utility. Basically, ISGT makes sure that its output x satisfies three conditions: 1) x is a TME in G'_T; 2) x is an NE in G_T; and 3) $\nexists (a_T, a_n) \in (A_T \setminus A'_T) \times A'_n$ such that $u_T(a_T, a_n) > u_T(x)$.

Our ISGT is shown in Algorithm 3.[1] Line 5 computes a TME for G'_T by solving Problem (2.2) (i.e., CoreTME), and Line 10 expands A'_i for each player i by solving the best response oracle. If 1) x is a TME in G'_T; and 2) x is an NE in G_T (by Theorem 3.1), the inner loop will terminate. Line 15 looks for a_T that gives a better utility to the team

[1]Another algorithm framework that we can develop is only using one loop. That is, we directly use the operations in Lines 15 and 16 of Algorithm 3 to replace the best response oracle for the team in Line 10 of Algorithm 3. However, experimental results show that this framework can be significantly slower than the current framework shown in Algorithm 3, which may be partially due to that this framework will add too many actions to the restricted game at early iterations. Therefore, we adopt the current framework.

Algorithm 3: ISG for a TME (ISGT)

```
1: Initialize $G'_T$ with $A' \leftarrow A'_T \times A'_n, u_T(x^*) \leftarrow -\infty, x^* \leftarrow \emptyset$
2: **repeat**
3:     Iteration $\leftarrow 1$
4:     **repeat**
5:         $x \leftarrow$ CoreTME($G'_T$)
6:         **if** $u_T(x) = u_T(x^*)$& Iteration $= 1$ **then**
7:             **return** $x^*$
8:         **end if**
9:         **for** $i \in N$ **do**
10:            $A'_i \leftarrow A'_i \cup \{a_i \leftarrow BR(x_{-i})\}$
11:        **end for**
12:        Iteration $\leftarrow$ Iteration $+1$
13:    **until** convergence ($A'$ does not change)
14:    $x^* \leftarrow x;$
15:    $A''_T \leftarrow \{a_T \mid u_T(a_T, a_n) > u_T(x), (a_T, a_n) \in (A_T \setminus A'_T) \times A'_n\}$
16:    $A'_T \leftarrow A'_T \cup A''_T$
17: **until** convergence ($A'_T$ does not change)
18: **return** $x$.
```

outside of G'_T and updates A'_T (Line 16). If no such a_T, the outer loop will terminate. Moreover, after obtaining an NE, ISGT records it as x^* in Line 14, and immediately returns it if there are not better NEs in Line 7.

Now ISGT definitely reduces the strategy space to $A_T \times A'_n$ to find a TME. To show that the output x of ISGT is a TME, we first show that if a TME in a restricted game is an NE in the full game, then it is also a TME in a larger restricted game including all adversary strategies.

Lemma 3.1. *If x is a TME in G'_T and an NE in G_T, then x is a TME in G''_T with $A'' = A'_T \times A_n$.*

Proof. Suppose that there is an NE $x'(i.e., (x'_T, x'_n))$ in G''_T such that $u_T(x') > u_T(x)$ and $\underline{A}_{n,x'_n} \not\subseteq A'_n$. Then, $u_n(x') \geq u_n(x'_T, x''_n)$ ($\forall x''_n$ with $\underline{A}_{n,x''_n} \subseteq A'_n$), i.e., $\min_{x''_n, \underline{A}_{n,x''_n} \subseteq A'_n} u_T(x'_T, x''_n) \geq u_T(x')$. Therefore, in G'_T, we have $\max_{x_T} \min_{x''_n} u_T(x_T, x''_n) \geq \min_{x''_n} u_T(x'_T, x''_n) \geq u_T(x') > u_T(x)$, which means that, $u_T(x)$ is not the TME value in G'_T, i.e., x is not a TME in the game G'_T, concluding the proof. □

Based on the above result, now we show that if a TME in a restricted game is an NE in the full game and there is no pure strategy profile outside the restricted game giving larger utility than this TME value, then it is also a TME in the full game.

Lemma 3.2. *x is a TME in G_T if a) x is a TME in G'_T, b) x is an NE in G_T, and c) $\nexists(a_T, a_n)$ such that $u_T(a_T, a_n) > u_T(x)$ with $a_T \in A_T \setminus A'_T$.*

Proof. By Lemma 3.1, x is a TME of G''_T with $A'' = A'_T \times A_n$. Suppose that there is a TME $x' = (x'_T, x'_n)$ in G_T such that $u_T(x') > u_T(x)$ and $\underline{A}_{T,x'_T} \not\subseteq A'_T$. That is, there is $a_i \notin A'_i (i \in T)$ such that $x'_i(a_i) > 0$. By the condition c), we have $u_T(a_T, a_n) \leq u_T(x)$ for each $a_T \in A_T \setminus A'_T$. Then, $u_T(a_i, x'_{-i}) \leq u_T(x) < u_T(x')$, and then there exists some a'_i such that $u_T(a'_i, x'_{-i}) > u_T(x')$ (otherwise, $u_T(x')$ will not be larger than $u_T(a_i, x'_{-i})$). Then we construct a new strategy x'' which is identical to x', but $x''_i(a_i) = 0$ and $x''_i(a'_i) = x'_i(a_i) + x'_i(a'_i)$. Then $u_T(x'') - u_T(x') = x'_i(a_i)(u_T(a'_i, x'_{-i}) - u_T(a_i, x'_{-i})) > 0$, which causes a contradiction, concluding the proof. □

Theorem 3.2. *x is a TME in G_T if: 1) x is a TME in G'_T; 2) x is an NE in G_T; and 3) $\nexists(a_T, a_n) \in (A_T \setminus A'_T) \times A'_n$ such that $u_T(a_T, a_n) > u_T(x)$.*

Proof. By Lemma 3.2, x is a TME in G''_T with $A''_T = (A_T, A'_n)$. Then, with the condition 2), x is a TME in G_T with $A = (A_T, A_n)$ by Lemma 3.1. □

In addition, ISGT (Line 7) indeed can terminate if the TME value does not change after adding all joint actions that are better than the equilibrium strategy by the following result.

Theorem 3.3. *x is a TME in G_T if x is a TME in G'_T and an NE in G_T, $u_T(x)$ is the TME value in G''_T with $\nexists(a_T, a_n) \in (A_T \setminus A'_T) \times A'_n$ such that $u_T(a_T, a_n) > u_T(x)$, and G'_T is a restricted game of G''_T that is a restricted game of G_T.*

Proof. x is an NE in G'_T because x is a TME in G'_T. Then, x is an NE in G''_T because x is an NE in G_T, and G'_T is a restricted game of G''_T that is a restricted game of G_T. Moreover, x is a TME in G''_T because $u_T(x)$ is the TME value in G''_T. By Theorem 3.2, x is a TME in G_T. □

More importantly, Propositions 3.3–3.5 have shown the impossibility of relaxing these conditions in Theorems 3.2 and 3.3. Now we can have our following conclusion.

Theorem 3.4. *ISGT converges to a TME in G_T.*

Proof. A is finite, so ISGT will terminate. ISGT terminates with output x satisfying conditions in Theorem 3.2 or 3.3, so x is a TME in G_T. □

3.2 The CTME Based ISGT (CISGT)

Even though ISGT can guarantee to converge to a TME, it is not efficient enough as ISGT needs to solve the non-convex Problem (2.2) in G'_T at each iteration. To speed up, we try to reduce the number of iterations by effectively initializing the restricted game G'_T through computing CTMEs. Now we first discuss the limitations of computing TMEs based on CTMEs.

3.2.1 Limitations of Computing TMEs by CTMEs

A CTME is close to a TME, and CTMEs are used to approximate TMEs [14] by Transforming the correlated strategy in a CTME into the team's Mixed Strategy Profile (TMSP). In this section, we show the limitations of computing TMEs based on CTMEs. We first show that using TMSPs may cause an arbitrarily large loss to the team. Second, we show that a TME in G'_T with the strategies for computing a CTME may not be an NE in G_T, and it may not be a TME in G_T even it is an NE in G_T. Moreover, we show that using the strategy profile in this TME of such G'_T may cause an arbitrarily large loss to the team as well.

To our best knowledge, the best algorithm [14] to obtain a TMSP is: Given the team's strategy in a CTME: $\overline{x}_T \in \Delta(A_T)$, player 1's mixed strategy $x_1(a_1) = \sum_{a' \in A_{T\setminus\{1\}}} \overline{x}_T(a_1, a')(\forall a_1 \in A_1)$; player i's mixed strategy $x_i(a_i) = \frac{1}{|\underline{A}_{i,\overline{x}_T}|}$ if $|\underline{A}_{i,\overline{x}_T}| > 0$, otherwise $x_i(a_i) = 0$ $(\forall i \in T \setminus \{1\}, a_i \in A_i)$. This algorithm returns the team's best TMSP among TMSPs obtained by exchanging player 1 with each team member. We call it TMSP-Alg. To measure the inefficiency of transforming the correlated strategies in a CTME into a TMSP, we adopt the concept of Price of Correlated strategies (PoC), introduced in Zhang and An (2020). Formally, PoC $= \frac{v_m}{v_c}$, where v_m is the TME value and v_c is the team utility obtained from a TMSP. Unfortunately, PoC can be arbitrarily large in G_T.

Proposition 3.6. *PoC can be arbitrarily large in G_T.*

Proof. Consider G_T with three players (two teammates), $A_1 = A_2 = A_3 = \{1, 2, 3\}$, and the following utility function:

$$u_T(a_1, a_2, a_3) = \begin{cases} 1 & \text{if } a_1 = a_2 = a_3 \\ \frac{1}{4} & \text{if } a_1 = 1, a_2 = 2 \\ -\frac{5}{4} & \text{if } a_1 = 2, a_2 = 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.5}$$

A CTME $\overline{x}$ is $\overline{x}_T(1,1) = \overline{x}_T(2,2) = \overline{x}_T(3,3) = \frac{1}{3}$ (note that, given any adversary strategy, one of these three pure strategies dominates other strategies, and any of them should be played with nonzero probability otherwise the adversary's best response gives utility 0 to the team) and $\overline{x}_n(1) = \overline{x}_n(2) = \overline{x}_n(3) = \frac{1}{3}$. Now, by TMSP-Alg, $\overline{x}$'s unique TMSP prescribes that $x_1 = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ and $x_2 = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, while the adversary (player 3) is indifferent between playing any strategies given this TMSP. Then, $v_c = 0$.

A TME is the pure strategy profile $(1, 2, 3)$ (obtained from solving Problem (2.2)), where player 1 plays action 1, player 2 plays action 2, while player 3 plays action 3. The team's utility is $u_T(1, 2, 3) = 0.25$. The reason is that the team will obtain utility 0 if any team member unilaterally deviates to other actions, and the adversary is indifferent among all strategies.

Therefore, PoC $= \frac{v_m}{v_c} = \frac{0.25}{0} = \infty$. □

Another idea is to compute a TME in G'_T with strategies for computing a CTME. There are two cases: 1) G'_T including all support sets of all players in a CTME; and 2) G'_T including all strategies for computing a CTME. However, in each case, this TME may not be an NE in G_T, and it may not be a TME in G_T even it is an NE in G_T by the following results.

Proposition 3.7. *x may not be an NE in G_T if $\overline{x}$ is a CTME in G_T, and x is a TME in G'_T with $A' = (\times_{i\in T}\underline{A}_{i,\overline{x}_T}) \times \underline{A}_{n,\overline{x}_n}$.*

Proof. Consider G_T with three players (two teammates), $A_1 = A_2 = \{1, 2\}$, $A_3 = \{1, 2, 3\}$, and the following utility function:

$$u_T(a)=\begin{cases} 10 & \text{if } a = (1,1,1) \text{ or } (2,2,2) \\ 10 & \text{if } a = (1,1,3) \text{ or } (2,2,3) \\ -10 & \text{if } a = (2,1,3) \text{ or } (1,2,3) \\ 0 & \text{otherwise} \end{cases} \tag{3.6}$$

A CTME $\overline{x}$ is $\overline{x}_T(1,1) = \overline{x}_T(2,2) = 0.5$ and $\overline{x}_3(1) = \overline{x}_3(2) = 0.5$ with utility 5 for the team (it is easy to verify that no players would like to deviate to other strategies, e.g., action 3 for the adversary with $u_T(\overline{x}_T, 3) = 10 > 5$ is not better than $\overline{x}_3$). Then we have G'_T with with $A'_1 = A'_2 = A'_3 = \{1, 2\}$. According to the analysis on the case in Eq.(3.2), x with $x_i(1) = x_i(2) = 0.5$ is a TME in G'_T with utility 2.5 for the team. However, for the adversary action 3, $u_T(x_T, 3) = 0.5\times0.5(10+10-10-10) = 0 < 2.5$. Therefore, x is not an NE in the original game G_T. □

Corollary 3.1. *x may not be an NE in G_T if $\overline{x}$ is a CTME in G_T and is computed in G'_T, and x is a TME in G'_T.*

Proof. Suppose a CTME $\overline{x}$ is computed in G'_T and $A' = (\times_{i\in T}\underline{A}_{i,\overline{x}_T}) \times \underline{A}_{n,\overline{x}_n}$. By Proposition 3.7, x may not be an NE in G_T, even if x is a TME in G'_T. □

Proposition 3.8. *x may not be a TME in G_T if $\overline{x}$ is a CTME in G_T, and x is a TME in G'_T with $A' = (\times_{i\in T}\underline{A}_{i,\overline{x}_T}) \times \underline{A}_{n,\overline{x}_n}$ and an NE in G_T.*

Proof. Consider the case in Eq.(3.2). A CTME $\overline{x}$ is $\overline{x}_T(1,1) = \overline{x}_T(2,2) = 0.5$ and $\overline{x}_3(1) = \overline{x}_3(2) = 0.5$ with utility 5 for the team (it is easy to verify that no players would like to deviate to other strategies). Then we have G'_T with with $A'_1 = A'_2 = A'_3 = \{1, 2\}$.

According to the analysis on the case in Eq.(3.2), x with $x_i(1) = x_i(2) = 0.5$ is a TME in G'_T with utility 2.5 for the team. x is an NE in G_T. However, according to the analysis on the case in Eq.(3.2), x is not a TME in G_T. □

Corollary 3.2. *x may not be a TME in G_T if $\overline{x}$ is a CTME in G_T and is computed in G'_T, and x is a TME in G'_T and an NE in G_T.*

Proof. Suppose a CTME $\overline{x}$ is computed in G'_T and $A' = (\times_{i\in T}\underline{A}_{i,\overline{x}_T}) \times \underline{A}_{n,\overline{x}_n}$. By Proposition 3.8, x may not be a TME in G_T, even if x is a TME in G'_T and an NE in G_T. □

Computing a TME in G'_T with strategies for computing a CTME cannot guarantee to obtain a TME in G_T but also can cause an arbitrarily large loss to the team.

Proposition 3.9. *If $\overline{x}$ is a CTME in G_T, and x is a TME in G'_T with $A' = (\times_{i\in T}\underline{A}_{i,\overline{x}_T}) \times \underline{A}_{n,\overline{x}_n}$, then playing x_T may cause an arbitrarily large loss to the team.*

Proof. Consider G_T with utilities shown in Eq.(3.6). As shown in the proof for Proposition 3.7, A CTME $\overline{x}$ is $\overline{x}_T(1,1) = \overline{x}_T(2,2) = 0.5$ and $\overline{x}_3(1) = \overline{x}_3(2) = 0.5$ with utility 5 for the team. Then we have G'_T with with $A'_1 = A'_2 = A'_3 = \{1,2\}$, and x with $x_i(1) = x_i(2) = 0.5$ is a TME in G'_T with utility 2.5 for the team. Given x_T, the adversary best response is action 3 with utility $u_T(x_T, 3) = 0.5 \times 0.5(10+10-10-10) = 0$ for the team. Now an NE $x' = ((\frac{1}{3}, \frac{2}{3}), (\frac{1}{3}, \frac{2}{3}), (\frac{2}{3}, 0, \frac{1}{3}))$ (it is easy to verify that no players would like to deviate to other strategies, e.g., action 3 for the adversary with $u_T(x_T, 2) = \frac{40}{9} > \frac{10}{9}$ is not better than x'_3) will given utility $\frac{10}{9}$ to the team. Then, playing x_T may cause an arbitrarily large loss to the team because $\frac{10/9}{0} = \infty$. □

Corollary 3.3. *If $\overline{x}$ is a CTME in G_T, and x is a TME in G'_T where $\overline{x}$ is computed, then playing x_T may cause an arbitrarily large loss to the team.*

Proof. Suppose a CTME $\overline{x}$ is computed in G'_T and $A' = (\times_{i\in T}\underline{A}_{i,\overline{x}_T}) \times \underline{A}_{n,\overline{x}_n}$. By Proposition 3.9, playing x_T may cause an arbitrarily large loss to the team. □

Algorithm 4: The CTME Based ISGT (CISGT)

```
1: Initialize $G'_T$ with $A' = A'_T \times A'_n, u_T(x^*) = -\infty, x^* \leftarrow \emptyset$
2: repeat
3:     $\overline{x} \leftarrow CoreCTME(G'_T)$;
4:     $A'_T \leftarrow A'_T \cup \{a_T \leftarrow BR(\overline{x}_n)\}$
5:     $A'_n \leftarrow A'_n \cup \{a_n \leftarrow BR(\overline{x}_T)\}$
6: until convergence ($A'$ does not change)
7: $x_T \leftarrow$TMSP-Alg($\overline{x}_T$)
8: if $u_T(x_T, BR(x_T)) \geq u_T(\overline{x})$ then
9:     return $(x_T, \overline{x}_n)$
10: end if
11: repeat
12:     Lines 4–13 in ISGT
13:     if $u_T(x) \geq u_T(\overline{x})$ then
14:         return $x$
15:     end if
16:     Lines 14–16 in ISGT
17: until convergence ($A'_T$ does not change)
18: return $x$.
```

3.2.2 CISGT: Efficiently Converging to a TME

Based on our discussion in the previous section, this section proposes our CISGT. In addition to the operations in ISGT, CISGT has two new operations: 1) it initializes the restricted game through computing a CTME to reduce the number of iterations for solving Prolbem (2.2); and 2) it exploits that the team's utility in a CTME is an upper bound of the TME value [14] to terminate earlier.

Our CISGT is shown in Algorithm 4.[1] CISGT uses Vanilla-ISG to compute a CTME at Lines 2–6. Then, CISGT computes a TMSP at Line 7 and then checks whether we have obtained a TME to return it at Lines 8–9. After that, CISGT repeats the operations in ISGT to compute a TME in G'_T and makes sure that it is also an NE in G_T at Line 12. CISGT then checks whether we have obtained a TME to return it at Lines 13–14. At Line 16, CISGT adds actions by repeating the operations in ISGT.

To show the convergence of CISGT, we first show that, a TMSP is part of a TME if the utility obtained by it is not less than the team's utility in a CTME.

[1]When we compute an ϵ-TME, we only need to set $u_T(x_T, BR(x_T)) \geq u_T(\overline{x}) - \epsilon$ at Line 8 and $u_T(x) \geq u_T(\overline{x}) - \epsilon$ at Line 13, and all properties still hold.

Theorem 3.5. *In G_T, given a CTME $\overline{x} = (\overline{x}_T, \overline{x}_n)$ and a mixed strategy profile for the team x_T such that $u_T(x_T, BR(x_T)) \geq u_T(\overline{x})$, then $(x_T, \overline{x}_n)$ is a TME.*

Proof. Let v^* be the TME value of G_T. By the relation between CTMEs and TMEs discussed by Basilico et al. (2017), we have $v^* \leq u_T(\overline{x}) = \max_{\overline{x}'_T} \min_{\overline{x}'_n} u_T(\overline{x}'_T, \overline{x}'_n) = \max_{\overline{x}'_T} u_T(\overline{x}'_T, \overline{x}_n)$. Then, we have $u_T(x_T, BR(x_T)) = \min_{x'_n} u_T(x_T, x'_n) \leq u_T(x_T, \overline{x}_n) \leq \max_{x'_i} u_T(x_{T\setminus\{i\}}, x'_i, \overline{x}_n) \leq \max_{\overline{x}'_T} u_T(\overline{x}'_T, \overline{x}_n) = u_T(\overline{x})$. Consequently, given x_T, for any adversary strategy x'_n, we have $u_n(x_T, x'_n) - u_n(x_T, \overline{x}_n) = -u_T(x_T, x'_n) - (-u_T(x_T, \overline{x}_n)) = u_T(x_T, \overline{x}_n) - u_T(x_T, x'_n) \leq u_T(\overline{x}) - u_T(x_T, BR(x_T)) \leq 0$. Similarly, given $\overline{x}_n$ and $x_{T\setminus\{i\}}$, for any player i's strategy x'_i, we have $u_T(x_{T\setminus\{i\}}, x'_i, \overline{x}_n) - u_T(x_T, \overline{x}_n) \leq u_T(\overline{x}) - u_T(x_T, BR(x_T)) \leq 0$. Then, $(x_T, \overline{x}_n)$ is an NE. In addition, due to $u_T(x_T, BR(x_T)) = \min_{x'_n} u_T(x_T, x'_n) \leq \max_{x'_T} \min_{x'_n} u_T(x'_T, x'_n) = v^*$, we have $u_T(x_T, \overline{x}_n) - v^* \leq u_T(\overline{x}) - u_T(x_T, BR(x_T)) \leq 0$ and $v^* - u_T(x_T, \overline{x}_n) \leq u_T(\overline{x}) - u_T(x_T, BR(x_T)) \leq 0$. Therefore, $(x_T, \overline{x}_n)$ is a TME. □

Similarly, a TME in G'_T is a TME in G_T if the utility obtained by it is not less than the team's utility in a CTME.

Theorem 3.6. *x is a TME in G_T if x is a TME in G'_T and an NE in G_T, $\overline{x}$ is a CTME in G_T with $u_T(x) \geq u_T(\overline{x})$.*

Proof. Let v^* be the TME value of G_T. By the relation between CTMEs and TMEs discussed by Basilico et al. (2017), we have $u_T(x) \leq v^* \leq u_T(\overline{x})$. Obviously, x is a TME in G_T. □

Finally, based on the above results, we have the following conclusion.

Theorem 3.7. *CISGT converges to a TME.*

Proof. First, CISGT will converge because the action set for each player is finite in G_T. Second, the output x is a TME by Theorems 3.4–3.6. □

3.3 Experiment Evaluation

We experimentally evaluate CISGT, comparing the performance of CISGT with that of ISGT and the state-of-the-art algorithm [16] (FullTME) for computing TMEs. FullTME enumerates all pure strategies and uses global optimization techniques to approximate multilinear terms in Problem (2.2) by a mixed-integer linear program. We use CPLEX solver (version 12.9) for solving all linear programs. All experiments are run on a machine with 6-core 3.6GHz CPU and 32GB memory.

We conduct experiments on the classic network security games[1] [20, 21, 70] to evaluate our approach. In a network security game, the adversary starts at a source node (he may have many possible source nodes) and travels along the path he chooses to one of his targets. That is, an action (a pure strategy) of the adversary is a path from a source to a target, and then the action space includes all possible paths. The police officers form a team, and each police occupies one of the possible edges on the network to try to catch the adversary before the target is reached. That is, an action (a pure strategy) of each team member is an edge. Similar to police officers in the NYPD, who maintain "sector integrity" [19], the action space of each police officer is disjoint with others in our setting. The adversary may have different values for different targets, and the adversary will succeed if his choosing path does not overlap with the edges chosen by the team; otherwise, the adversary will obtain nothing. All networks are generated by the grid model with random edges [71], which models the real urban network with some parameters. That is, it samples a square network with $L \times W$ nodes, and it generates horizontal/vertical edges between neighbors with probability p, and diagonal ones with probability q (usually, $p \geq q$ in real networks [71]). By default, $n = 3$, and results are all averaged over 30 instances that are randomly generated.

Vanilla-ISG to compute CTMEs is the double oracle algorithm proposed by Jain et al. (2011), including the linear program for computing a maximin strategy for the team, and mixed-integer linear programs of best response oracles for the team and the adversary, respectively. These best response oracles can be easily extended to CISGT.

[1]Network security games can be easily extended to other games, including the robot planning problem in the adversary environment [5], patrolling games with alarm systems [68], and green security games [46]. Then our results will also hold in these games.

L×W	5×5	5×5	5×5	5×5	5×5	4×4	6×6	8×8	10×10
(p,q)	(0.8,0.6)	(0.7,0.5)	(0.6,0.4)	(0.5,0.3)	(0.4,0.2)	(0.4,0.2)	(0.4,0.2)	(0.4,0.2)	(0.4,0.2)
FullTME		∞	448s	50.4s	17.8s	0.3s	∞		
ISGT					>1000s	4s	>1000s		
CISGT	9.8s	5.9s	4.7s	3.7s	2.3s	2.2s	8.3s	24s	57s

TABLE 3.1: Computing TMEs: ∞ represents out of memory.

L×W	4×4	4×4	4×4	4×4	4×4	5×5	6×6	7×7	8×8
(p,q)	(0.4,0.2)	(0.3,0.2)	(0.3,0.1)	(0.2,0.1)	(0.1,0,1)	(0.1,0.1)	(0.1,0.1)	(0.1,0.1)	(0.1,0.1)
Vanilla-ISG	35%	38%	38%	30%	26%	53%	47%	56%	55%
TMSP-Alg	61%	50%	51%	44%	38%	75%	54%	79%	87%

TABLE 3.2: Gaps relative to CISGT.

TMEs for restricted games are computed by the algorithm proposed by Zhang and An (2020), i.e., FullTME computes TMEs in restricted games.

Results in Table 3.1 show that CISGT is orders of magnitude faster than ISGT and FullTME.[1] Specially, when FullTME, enumerating all pure strategies to compute a TME in a full game, runs out of the memory, CISGT still runs efficiently.

In addition, we compare the solution quality of CISGT with that of Vanilla-ISG and TMSP-Alg. Table 3.2 shows the possible gaps, which are the relative distance between the team utility (v_m) obtained by CISGT and the team utility v obtained by Vanilla-ISG or TMSP-Alg, i.e., $\frac{|v-v_m|}{|v_m|} \times 100\%$. The team will lose more utility if the gap is larger. We can see that the team may lose a large utility if Vanilla-ISG or TMSP-Alg is deployed.

3.4 Chapter Summary

This chapter proposes an efficient ISG algorithm (CISGT) to compute TMEs for zero-sum multiplayer games. We first study the properties of ISG for multiplayer games, showing that ISG converges to an NE but may not converge to a TME. Second, we design ISGT by exploiting the property of TMEs and show that ISGT converges to a TME and the impossibility of relaxing its conditions. Third, to further improve the scalability, we design CISGT by initializing the strategy space of ISGT via computing

[1]In addition, ISGT's another framework we mentioned in Section 3.1.3 needs 898s on the smallest network with 4×4 and (0.4,0.2).

a CTME. Finally, experimental results show that CISGT is orders of magnitude faster than ISGT and the state-of-the-art algorithm in large games.

Chapter 4

Computing Correlated Team-Maxmin Equilibria for Optimal Escape Interdiction on Transportation Networks

This chapter[1] focuses on computing CTMEs [14] in normal-form games for optimal escape interdiction on transportation networks. This chapter uses the CTME as the solution concept for the novel escape interdiction game model that captures the interaction between an escaping adversary (attacker) and a team of multiple defense resources (defender) with correlated time-dependent strategies on transportation networks. We first introduce a novel Escape Interdiction Game (EIG) to model time-dependent strategies for both players, where the defender chooses a sequence of intersections for each police officer to protect, while the attacker chooses an escape path and his travel time on different edges along the path. To overcome the computational challenge, we first propose our solution, *EIGS* (an ISG variant for this scenario), which incorporates the following key components: 1) a new double oracle structure to avoid calling the hard oracle, i.e., the best response attacker oracle, frequently; 2) novel mixed-integer linear programming

[1]The work in this chapter has been published as **Youzhi Zhang**, Bo An, Long Tran-Thanh, Zhen Wang, Jiarui Gan, Nicholas R. Jennings. Optimal escape interdiction on transportation networks. *Proceedings of the 26th International Joint Conference on Artificial Intelligence* (**IJCAI**), pp.3936-3944, 2017.

(MILP) formulations to compute the best response strategies for both players; and 3) a greedy algorithm and an efficient modified Dijkstra algorithm to efficiently generate improving strategies. To further improve the scalability to solve large-scale scenarios with hundreds of intersections, we introduce a novel defender algorithm *RedAD* that reduces the strategy space by deploying each defender resource to protect only one node, assuming that the attacker travels with maximum speed, and using a novel algorithm to contract the graph before using the MILP to compute the attacker response strategy. We conduct extensive experiments showing that *EIGS* can efficiently obtain a robust solution significantly outperforming existing approaches in problems of related small scales and *RedAD* can scale up to realistic-sized transportation networks of hundreds of intersections with good solution quality.

4.1 Modelling Traffic Network

In this section, we introduce a realistic macroscopic traffic model to illustrate how the traffic flow influences the travel time in a transportation network [72–75].

Network Structure: The urban road network is modeled as a directed graph $G = (V, E)$ consisting of a set of directed links E to represent the roads, and a set of nodes V to represent the intersections. For each link $e = (\sigma_e, \tau_e)$, the traffic flows from its start node σ_e to its end node τ_e. An extra node $v_\infty \in V$ represents the external world, which is both the source of the flow entering the network and the sink of flow exiting it. In particular, the sets of entry and exit links are denoted by $E_{entry} = \{e|\sigma_e = v_\infty\}$ and $E_{exit} = \{e|\tau_e = v_\infty\}$, respectively. Accordingly, the remaining subset of the links is the set of internal links $E_{inter} = \{e|\sigma_e, \tau_e \in V \setminus \{v_\infty\}\}$. It assumes that there are no self-loops, i.e., $\sigma_e \neq \tau_e$ for each link $e \in E$. For each link $e \in E$, T_e is the minimal travel time when the road is empty, C_e is the link capacity, and the sets of upstream links and downstream links are denoted by $E_e^- = \{e'|\tau_{e'} = \sigma_e\}$ and $E_e^+ = \{e'|\sigma_{e'} = \tau_e\}$, respectively.

Traffic Flow: Each entry link $e \in E_{entry}$ is associated with a constant traffic demand $d_e \geq 0$ modelling the rate of vehicles entering the node τ_e from the external world[1]. Traffic flows among consecutive links according to a static turning preference matrix $R \in \mathbb{R}_+^{E \times E}$ whose entry $R_{ee'}$ represents the fraction of vehicles flowing out of link e that are routed to link e'. These turning ratios model the route choice behavior of drivers, justified by empirical observations [76]. Conservation of mass implies that $\sum_{e' \in E_e^+} R_{ee'} = 1$ for all $e \in E$. Moreover, the natural topological constraints imply that $R_{ee'} = 0$ if $\tau_e \neq \sigma_{e'}$. To avoid trivial cases, we assume that for every node v, there exists at least one directed path from v to the node v_∞, so that every vehicle will eventually exit. The rate of vehicles passing link e is denoted by traffic flow f_e. According to [75], there exists a unique *equilibrium flow* $\mathbf{f} = \{f_e, e \in E\}$ such that:

$$f_e = d_e \qquad \forall e \in E_{entry} \tag{4.1}$$

$$f_e = \sum_{e' \in E_e^-} f_{e'} R(e', e) \qquad \forall e \in E_{inter} \cup E_{exit} \tag{4.2}$$

For any given demand vector $\mathbf{d}$ and turning preference matrix R, the induced equilibrium flow $\mathbf{f}$ is used to calculate the minimal travel time t_e on link e. This solution can be easily extended to handle more complex traffic models [72, 73, 77].

Travel Time: t_e on a link e, strictly increasing with f_e, is defined by the commonly used Bureau of Public Roads (BPR) function [74, 78]:

$$t_e = T_e(1 + 0.15 \times (f_e/C_e)^4) \tag{4.3}$$

4.2 The Escape Interdiction Game

The EIG is played between the escapee (attacker), and the police officers (defender). The attacker tries to escape to the external world v_∞ via any exit node $d \in D \subset \{v | \exists e = (v, v_\infty) \in E\}$ after conducting some criminal activity at node $v_0^a \in V$. There are m defender resources (police officers/teams), initially located at intersections $v_0^1, \ldots, v_0^m \in$

[1]This rate is possibly time-varying. For simplicity we consider it as constant during the relatively short time period of interest.

V. To only consider nontrivial problems, we assume that $v_0^a \neq v_0^r$, for all $r \in \mathcal{R} = \{1, \dots, m\}$. W.l.o.g., we assume that the event starts at time 0 and ends at time $t_{max} > 0$. The traffic flow on each edge e is likely to affect the corresponding travel time t_e of players, which is measured by Eq.(4.3). The traffic network, the initial positions of the bank and the police officers are common knowledge, but both players cannot see each other until the attacker gets caught.

Strategies: A pure attacker strategy is a sequence of states $A = \langle a_1 = (v_0^a, 0), \dots, a_j = (v_j, t_j^a), \dots, a_{k-1} = (d \in D, t_{k-1}^a), a_k = (v_\infty, t_k^a \leq t_{max})\rangle$, where each state $a_j = (v_j, t_j^a)$ represents the situation that the attacker arrives at node v_j at time t_j^a. For every two consecutive states $a_j = (v_j, t_j^a)$ and $a_{j+1} = (v_{j+1}, t_{j+1}^a)$ in an attacker strategy A, it is satisfied that $v_{j+1} \in N(v_j)$, where $N(v_j) = \{u \in V | (v_j, u) \in E\}$, and $t_{j+1}^a \geq t_j^a + t_{(v_j, v_{j+1})}$. This means the attacker can spend any time longer than or equal to the minimal travel time $t_{(v_j, v_{j+1})}$ and makes the attacker's strategy space $\mathcal{A}$ continuous. A mixed attacker strategy is denoted by $\mathbf{y} = \langle y_A \rangle$, with y_A representing the probability that A is played.

A state of defender resource d_r is a tuple $s^r = (v^r, t^{r,in}, t^{r,out})$, representing the situation that d_r is protecting node v^r during $[t^{r,in}, t^{r,out}]$. We assume that d_r can only stay at a node for a multiple of a constant time interval δ (e.g., 10 seconds), i.e., $t^{r,out} - t^{r,in} = \kappa\delta$, where κ is a nonnegative integer. To simplify our analysis, we assume that d_r captures the attacker at a node. Thus, for every two consecutive states $s_i^r = (v_i^r, t_i^{r,in}, t_i^{r,out})$ and $s_{i+1}^r = (v_{i+1}^r, t_{i+1}^{r,in}, t_{i+1}^{r,out})$, $t_{i+1}^{r,in} - t_i^{r,out} = dist(v_i^r, v_{i+1}^r)$, which is the minimal travel time between v_i^r and v_{i+1}^r. This can easily be computed using Dijkstra's shortest path algorithm.

A pure strategy for the defender is a set of m schedules, i.e., $S = \{S^r : r \in \mathcal{R}\}$. The schedule for d_r is a sequence of states $S^r = \langle s_1^r, \dots, s_i^r, \dots, s_k^r \rangle$, where $s_1^r = (v_0^r, 0, t_1^{r,out})$ and $s_k^r = (v_k^r, t_k^{r,in}, t_{max})$. Moreover, d_r cannot drive to the external world v_∞ to capture the attacker, i.e., $v_i^r \in V \setminus \{v_\infty\}, \forall s_i^r \in S^r$. The defender's pure strategy space is denoted by $\mathcal{S}$. A defender's mixed strategy is $\mathbf{x} = \langle x_S \rangle$, with x_S representing the probability that S is played.

Utility: We are only concerned with whether the attacker can be captured or not, so we assume a zero-sum game. For $a_j = (v_j, t^a_j)$ and $s^r_i = (v^r_i, t^{r,in}_i, t^{r,out}_i)$, we say the attacker is captured by d_r at node v^r_i if $v^r_i = v_j$ and $t^{r,in}_i \leq t^a_j \leq t^{r,out}_i$. In such a case, $z_{s^r_i,a_j} = 1$, otherwise $z_{s^r_i,a_j} = 0$. If an attacker successfully escapes to the external world, he gains a utility of 1 and the defender gets -1; otherwise both players get a utility of 0. Formally, the defender utility is given by:

$$U_d(S,A)=\begin{cases}0 & \text{if } \exists z_{s^r_i,a_j}=1, a_j\in A, s^r_i\in S^r, r\in\mathcal{R}\\ -1 & \text{otherwise}\end{cases} \tag{4.4}$$

Given $\mathbf{x}$ and A, the expected utility of the defender is $U_d(\mathbf{x}, A) = \sum_{S\in\mathcal{S}} U_d(S, A)x_S$. Accordingly, we can define $U_d(\mathbf{x}, \mathbf{y}) = \sum_{A\in\mathcal{A}} y_A U_d(\mathbf{x}, A)$. Note that $U_a = -U_d$.

Equilibrium: We use the NE (i.e., CTME) as the solution concept of this game as the defender and the attacker players decide their strategies simultaneously. Thus, the optimal mixed strategy $\mathbf{x}$ of the defender can be computed by solving the following linear program (*LP*).

$$\max \quad U^* \tag{4.5}$$

$$\text{s.t.} \quad U^* \leq U_d(\mathbf{x}, A) \qquad \forall A \in \mathcal{A} \tag{4.6}$$

$$\sum\nolimits_{S\in\mathcal{S}} x_S = 1, x_S \geq 0 \qquad \forall S \in \mathcal{S} \tag{4.7}$$

4.3 Solution Approach

Solving the *LP* in Eqs.(4.5)–(4.7) under the exponentially large $\mathcal{S}$ and the continuous $\mathcal{A}$ is challenging. To handle this challenge, we develop a novel double oracle algorithm *EIGS* (EIG Solver). Now, we first show that solving EIG is NP-hard.

Theorem 4.1. *Computing an NE of EIG is NP-hard.*

Proof. We reduce 3-SAT to the computation of an NE of EIG. 3-SAT asks if there exists an assignment of values to a set of boolean variables that satisfies a given boolean

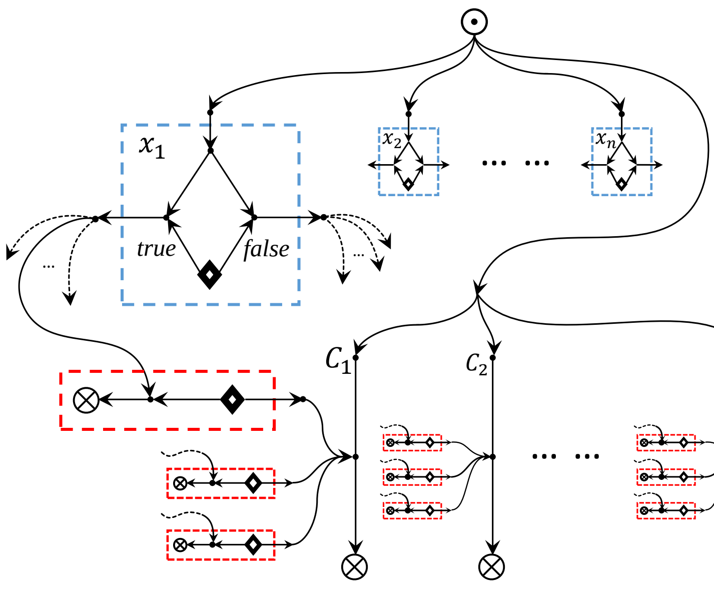


FIGURE 4.1: Reduce 3-SAT to EIG.

formula in 3CNF (i.e., a conjunctive normal form where each clause is limited to at most three literals). Let the variables of the 3-SAT be $x_1, \dots, x_n$, and the clauses be $C_1 \wedge C_2 \wedge \cdots \wedge C_k$. We construct an EIG on a road network shown in Figure 4.1, such that, as we will show later, the 3CNF is satisfiable if and only if (denoted as $\Leftrightarrow^1$) there exists a defender pure strategy intercepting *all* attacker pure strategies if and only if (denoted as $\Leftrightarrow^2$) the attacker gets caught with probability 1 in an NE (under the mixed strategy setting). Particularly, we will focus on the proof on "$\Leftrightarrow^1$" as "$\Leftrightarrow^2$" is trivial. If there exists such a defender pure strategy, then the defender can simply play a mixed strategy incorporating only this pure strategy regardless of which strategy the attacker plays. Otherwise, for any defender strategy, the attacker can always find a pure strategy not being intercepted by at least one pure strategy in the support of the defender strategy, which gives the attacker a non-zero probability of escaping successfully.

In this graph, the $\odot$ on the top represents the source node v_0^a where the attacker begins to escape; each of the $\bigotimes$'s on the bottom and in the red boxes represents an exit node through which the attacker can escape to the external world; each diamond represents a police station, where one police car is available.

For each variable x_i of the 3-SAT problem, we create a gadget shown in the blue box, in which there is a police station with two outbound roads. The police car from this station can choose either to move left or right, corresponding to $x_i = true$ or $false$, respectively. The choice of directions of all the police cars in these blue boxes corresponds to an assignment of values to $x_1, \dots, x_n$ in the 3-SAT problem.

For each clause C_j, we create a gadget consisting of a road, call it the clause road, from v_0^a to an exit node and three components each corresponding to a literal of the clause. In each of the red boxes, there is a police station with two outbound roads. The one to the right links to the road of the clause, so that choosing this direction, the police car can intercept attacker pure strategies passing through the clause road. The one to the left leads to an exit node, and joints in the middle with another road coming from one of the blue boxes and corresponding to: $x_i = true$ if the literal is $\neg x_i$; and $x_i = false$ if the literal is x_i. Namely, for example, if in the blue box x_i is chosen to be $true$, the $false$ direction is left open for the attacker to pass through, so that the police cars in the red boxes corresponding to $\neg x_i$ have to move left to intercept the attacker. In such a way, attacker pure strategies passing through a clause road can be intercepted (without missing any attack pure strategy exiting from the red boxes) only when one of its literals is made $true$ by choices of directions in the blue boxes.

We show that: the 3CNF is satisfiable $\Leftrightarrow$ there exists a defender pure strategy intercepting all attacker pure strategies.

1. The "$\Rightarrow$" direction. Suppose there exists a satisfying assignment for the 3-SAT problem. What the defender can do is to let the police car in each blue box move to the direction that is the same as the value of x_i in the satisfying assignment and stay at the first intersection. The other direction in each blue box is left open, but the roads passing through it eventually lead to the red boxes and can be intercepted by moving police cars in the red boxes to the left. Finally, since each clause contains at least one literal which is made true, the police cars in the corresponding red box do not need to move left to intercept the attacker as the attacker is already intercepted in the blue boxes; therefore, letting these police cars move to the clause road, all attacker pure strategies passing through the clause roads get intercepted as well.

2. The "$\Leftarrow$" direction. If there is not a satisfying assignment, then no matter how the defender moves the resources, there exists at least one clause such that the police cars

Algorithm 5: *EIGS*

```
1  Initialize $\mathcal{S}'$, $\mathcal{A}'$.
2  repeat
3      $(\mathbf{x},\mathbf{y}) \leftarrow CoreLP(\mathcal{S}',\mathcal{A}')$;
4      $\mathcal{S}^* \leftarrow \{betterDO(\mathbf{y})\}$;
5      if $\mathcal{S}^* = \emptyset$ then
6          $\mathcal{S}^* \leftarrow \{bestDO(\mathbf{y})\}$;
7      $\mathcal{S}' \leftarrow \mathcal{S}' \cup \mathcal{S}^*$;
8      $\mathcal{A}^* \leftarrow \{betterAO(\mathbf{x})\}$;
9      if $\mathcal{A}^* = \emptyset \wedge \mathcal{S}^* = \emptyset$ then
10         $\mathcal{A}^* \leftarrow \{bestAO(\mathbf{x})\}$;
11     $\mathcal{A}' \leftarrow \mathcal{A}' \cup \mathcal{A}^*$;
12 until convergence;
13 return $(\mathbf{x},\mathbf{y})$.
```

in the red boxes of its literals all have to move left to intercept the incoming attacker. The clause road is left open. Then the attacker strategy passing through the clause road reaches the exit node successfully. □

4.3.1 A Novel Double Oracle Framework

Now we develop *EIGS*, which first computes the equilibrium strategy for a significantly smaller restricted game, then iteratively computes improving strategies for both players, and finally converges to a global equilibrium.

EIGS is sketched in Algorithm 5. Line 1 initializes $\mathcal{S}'$ and $\mathcal{A}'$, which are small sets of pure strategies for both players. Then, *CoreLP* computes the maximin strategies $\mathbf{x}$ and $\mathbf{y}$ for the restricted game $\langle \mathcal{S}', \mathcal{A}' \rangle$ (Line 3) by solving the *LP* in Eqs.(4.5)–(4.7). To improve both players' utilities, the defender oracle (DO) (Lines 4–7) and the attacker oracle (AO) (Lines 8–11) are repeatedly used to find other strategies out of $\langle \mathcal{S}', \mathcal{A}' \rangle$ until no improving strategy can be found (Line 12). DO first calls efficient *betterDO* (better response DO) trying to find an improving strategy that may be suboptimal. If *betterDO* fails, it proceeds to *bestDO* (best response DO) that is optimal. Similarly, AO first calls efficient *betterAO* (better response AO), and if *betterAO* and DO both fail, it proceeds to *bestAO* (best response AO). This structure at Line 9 is different from the classic double oracle employed in security games [25, 26], where the best oracle will be called if the

Algorithm 6: *betterDO* $(\mathbf{y})$

```
$\overline{\mathcal{A}} \leftarrow \mathcal{A}'; \forall r \in \mathcal{R}: S^r \leftarrow \langle (v^r_0, 0, 0) \rangle, L^r \leftarrow 1, t^{r,out}_1 \leftarrow 0;$
while $\overline{\mathcal{A}} \neq \emptyset$ do
    for $r \in \mathcal{R}$ do
        for $v \in V \setminus \{v_\infty\}$ do
            $t[r][v] \leftarrow t^{r,out}_{L^r} + dist(v_{L^r}, v), \ \Delta y[r][v] \leftarrow 0;$
            for $A \in \overline{\mathcal{A}}$ do
                if $\exists a_j = (v, t^a_j) \in A, t[r][v] \leq t^a_j$ then
                    $\Delta y[r][v] \leftarrow \Delta y[r][v] + y_A;$
        $v^{r,*} \leftarrow \arg\max_v \Delta y[r][v];$
    $r^* \leftarrow \arg\max_r \Delta y[r][v^{r,*}];$
    if $\Delta y[r^*][v^{r,*}] > 0$ then
        $L^{r^*} \leftarrow L^{r^*} + 1, v_{L^{r^*}} = v^{r,*};$
        for $A \in \overline{\mathcal{A}}$ do
            if $\exists a_j = (v_{L^{r^*}}, t^a_j) \in A$ then
                $t^{r^*,out}_{L^{r^*}} \leftarrow \max\{t[r^*][v_{L^{r^*}}], t^a_j\}, \overline{\mathcal{A}} \leftarrow \overline{\mathcal{A}} \setminus \{A\};$
        $\kappa \leftarrow \lceil (t^{r^*,out}_{L^{r^*}} - t[r^*][v_{L^{r^*}}])/\delta \rceil, t^{r^*,out}_{L^{r^*}} \leftarrow t[r^*][v_{L^{r^*}}] + \delta \cdot \kappa;$
        $S^{r^*} \leftarrow S^{r^*} \cup \langle (v_{L^{r^*}}, t[r^*][v_{L^{r^*}}], t^{r^*,out}_{L^{r^*}}) \rangle;$
    else
        break;
return $\{S^r, r \in \mathcal{R}\}$.
```

corresponding better oracle fails. However, it is not efficient if one runs much slower than the other between best oracles.

4.3.2 Defender Oracle

The defender needs to find an improving path with a stop time such that it can intercept as many attacker paths as possible, i.e., increase the probability of catching the attacker. Formally, given $(\mathbf{x}, \mathbf{y})$ provided by *CoreLP*, S is added to $\mathcal{S}'$ if $U_d(S, \mathbf{y}) > U_d(\mathbf{x}, \mathbf{y})$. This can be formulated as an MILP, i.e., ***bestDO***. It first constructs a path attaching travel time and stop time, and then checks if it can intercept the given attacker path, i.e., both players meet at the same time and the same node. If there are more attacker paths being intercepted by this new defender path, the defender will get a higher utility. This procedure is represented by the following MILP:

$$\max -\sum\nolimits_{A\in\mathcal{A}'}(1-z_A)y_A \tag{4.8}$$

$$\text{s.t. } s^r_{1,v^r_0}=1, \sum\nolimits_{v\in V\setminus\{v_\infty\}} s^r_{i,v}=1 \quad \forall r,i \tag{4.9}$$

$$\omega_{r,i,(v,u)} \leq \min(s^r_{i,v}, s^r_{i+1,u}) \quad \forall r,i,v,u \tag{4.10}$$

$$\omega_{r,i,(v,u)} \geq s^r_{i,v}+s^r_{i+1,u}-1 \quad \forall r,i,v,u \tag{4.11}$$

$$t^{r,in}_1=0, t^{r,out}_{L^d_{max}}=t_{max}, t^{r,out}_i=t^{r,in}_i+\kappa_{r,i}\delta \ \forall r,i \tag{4.12}$$

$$t^{r,in}_{i+1}=t^{r,out}_i+\sum\nolimits_{v,u\in V\setminus\{v_\infty\}} dist(v,u)\omega_{r,i,(v,u)} \ \forall r,i \tag{4.13}$$

$$-M\alpha^{A,j}_{r,i} \leq t^{r,in}_i - t^A_j \leq M(1-\alpha^{A,j}_{r,i}) \ \forall r,i,A,j \tag{4.14}$$

$$-M\beta^{A,j}_{r,i} \leq t^A_j - t^{r,out}_i \leq M(1-\beta^{A,j}_{r,i}) \ \forall r,i,A,j \tag{4.15}$$

$$\gamma^{A,j}_{r,i} \leq (\alpha^{A,j}_{r,i}+\beta^{A,j}_{r,i}+s^r_{i,v^A_j})/3 \quad \forall r,i,A,j \tag{4.16}$$

$$\gamma^{A,j}_{r,i} \geq \alpha^{A,j}_{r,i}+\beta^{A,j}_{r,i}+s^r_{i,v^A_j}-2 \quad \forall r,i,A,j \tag{4.17}$$

$$z_A \leq \sum\nolimits_{j,r,i}\gamma^{A,j}_{r,i} \quad \forall A \tag{4.18}$$

$$s^r_{i,v}, \omega_{r,i,(v,u)}, \alpha^{A,j}_{r,i}, \beta^{A,j}_{r,i}, \gamma^{A,j}_{r,i}, z_A \in \{0,1\} \tag{4.19}$$

$$\kappa_{r,i}\in\mathbb{Z}_{\geq 0}, t^{r,in}_i, t^{r,out}_i \in [0,t_{max}] \tag{4.20}$$

Here, $s^r_{i,v}=1$ indicates that d_r arrives at v by the i-th state of S^r. Eq.(4.9) indicates that d_r starts at v^r_0 and stays on only one node in any state of S^r. In Eqs.(4.10) and (4.11), the $0/1$ variable $\omega_{r,i,(v,u)}$ encodes whether there is a path v-u between the i-th state and $(i+1)$-th state of S^r. Eq.(4.12) forces the strategy to start at time 0 and end at t_{max}, where L_{max} is a sufficiently large number to estimate the maximum length allowed of the defender strategy sequence, and the time that d_r stays on the i-th state is $\kappa_{r,i}\delta$. Eq.(4.13) ensures that d_r travels one stop to another through the shortest path. With Eqs.(4.14)–(4.18), z_A determines whether the attacker using A is caught. Specifically, (v^A_j, t^A_j) represents the attacker's j-th state in A; $\gamma^{A,j}_{r,i}$ indicates whether the attacker using A, meets d_r at his j-th state's position while d_r is at her i-th stop; $\alpha^{A,j}_{r,i}$ (respectively, $\beta^{A,j}_{r,i}$) indicates whether the attacker arrives at his j-th state's position after d_r arrives (respectively, before d_r leaves); and M is a very large number.

To speed up the above MILP, we propose a faster oracle ***betterDO*** as shown in Algorithm 6. It is a greedy algorithm, i.e., at each step, the defender goes to the nodes

Algorithm 7: *betterAO* (x)

```
1  $\forall v \in V \setminus \{v_0^a\}$, $\Delta x[v] \leftarrow \infty$, $t^a[v] \leftarrow \infty$;
2  $\Delta x[v_0^a] \leftarrow 0$, $t^a[v_0^a] \leftarrow 0$, $\overline{\mathcal{S}}_{v_0^a} \leftarrow \mathcal{S}'$, $Q \leftarrow V$;
3  while $Q \neq \emptyset$ do
4      $v \leftarrow \arg\min_{u \in Q} \Delta x[u]$;
5      if $v = v_\infty$ then
6          break;
7      $Q \leftarrow Q \setminus \{v\}$;
8      for each neighbor $u$ of $v$ do
9          $t \leftarrow \arg\min_{t' \geq t^a[v] + t_{(v,u)}} \Phi(t') := \sum_{S \in \overline{\mathcal{S}}_v} x_S \cdot \phi_u(S, t')$;
10         if $\Delta x[v] + \Phi(t) < \Delta x[u]$ then
11             $\Delta x[u] \leftarrow \Delta x[v] + \Phi(t)$;
12             $t^a[u] \leftarrow t^a[v] + t$;
13             $prev[u] \leftarrow v$;
14             $\overline{\mathcal{S}}_u \leftarrow \overline{\mathcal{S}}_v \setminus \{S | u \in S, t^a[u] \in [t^{in}_{S,u}, t^{out}_{S,u}]\}$;
15 $A \leftarrow \langle (v_0, 0), (v_1, t^a[v_1]), \ldots, (v_\eta, t^a[v_\eta]) \rangle$ such that $v_0 = v_0^a$, $v_\eta = v_\infty$, and
     $v_i = prev[v_{i+1}] \, \forall\, i = 0, \ldots, \eta - 1$;
16 return $A$.
```

with the highest marginal value. This marginal value on each node is the probability of intercepting the remaining attacker paths if the defender moves to that node. As shown in Algorithm 6, $\overline{\mathcal{A}}$ stores the attacker's paths that are not caught by any S^r. Initially, d_r is at her initial position v_0^r (Line 1). Lines 2–19 are the main loop, where we repeatedly allocate police officers to interdict the attacker's paths with the highest marginal value. Specifically, Lines 3–9 enumerate all the combinations $\langle r, v\rangle \in \mathcal{R} \times V$, and check how much value d_r can obtain if she moves to v and stays there as long as possible (therefore, in Lines 6–8, all attacker paths in which the attacker arrives at v later than $t[r][v]$ are intercepted, and the corresponding values are added to $\Delta y[r][u]$). d_{r^*} who can obtain the highest value among defender resources is chosen (Line 10). The remaining part is to update the finalized movement (Lines 13–15), where the attacker paths caught by the movement are removed from $\overline{\mathcal{A}}$, and $t^{r^*,out}_{L^{r^*}}$ is updated to the time that this movement can intercept as many attacker paths as possible. d_{r^*} stays at a node for $\kappa\delta$ (Line 16) and then updates her new state to the strategy sequence (Line 17).

4.3.3 Attacker Oracle

The attacker needs to find an improving path with a travel time such that it can minimize the probability of being caught. Formally, given $(\mathbf{x}, \mathbf{y})$ provided by *CoreLP*, A is added to $\mathcal{A}'$ if $U_a(\mathbf{x}, A) > U_a(\mathbf{x}, \mathbf{y})$. This can be formulated as an MILP, i.e., ***bestAO***. It first constructs a path attaching travel time, and then checks if it is intercepted by the given defender path, i.e., both players meet at the same time and the same node. If there are more defender paths intercepting this new attacker path, the attacker will get a lower utility. This procedure is represented by the following MILP:

$$\max \sum\nolimits_{S\in\mathcal{S}'}(1-z_S)x_S \tag{4.21}$$

$$\text{s.t. } A_{1,v_0^a}=1, A_{L^a_{max},v_\infty}=1, \sum\nolimits_{v\in V} A_{j,v}=1 \quad \forall j \tag{4.22}$$

$$A_{j+1,v_\infty} \geq A_{j,v_\infty} \quad \forall j \tag{4.23}$$

$$\sum\nolimits_{u\in N(v)} A_{j+1,u} \geq A_{j,v} \; \forall v\in V, j \tag{4.24}$$

$$\omega_{j,(v,u)} \leq \min(A_{j,v}, A_{j+1,u}) \quad \forall (v,u)\in E, j \tag{4.25}$$

$$\omega_{j,(v,u)} \geq A_{j,v} + A_{j+1,u} - 1 \quad \forall (v,u)\in E, j \tag{4.26}$$

$$t_1^a = 0 \tag{4.27}$$

$$t_{j+1}^a \geq t_j^a + \sum\nolimits_{(v,u)\in E} t_{(v,u)}\omega_{j,(v,u)} \quad \forall j \tag{4.28}$$

$$-M\alpha^j_{S,r,i} \leq t_i^{r,in} - t_j^a \leq M(1-\alpha^j_{S,r,i}) \; \forall S,r,i,j \tag{4.29}$$

$$-M\beta^j_{S,r,i} \leq t_j^a - t_i^{r,out} \leq M(1-\beta^j_{S,r,i}) \; \forall S,r,i,j \tag{4.30}$$

$$z_S \geq \alpha^j_{S,r,i} + \beta^j_{S,r,i} + A_{j,v_i^{Sr}} - 2 \quad \forall S,r,i,j \tag{4.31}$$

$$A_{j,v}, \alpha^j_{S,r,i}, \beta^j_{S,r,i}, \omega_{j,(v,u)}, z_S \in \{0,1\}, t_j^a \in [0, t_{max}] \tag{4.32}$$

Here $A_{j,v}$ indicates whether the attacker arrives at v in the j-th state. Eq.(4.22) ensures that the attacker starts at v_0^a, terminates at v_∞, and only arrives at one node in each state. Eqs.(4.23)–(4.24) fix the attacker at the sink node v_∞ in the following states after he reaches v_∞ and ensure that the attacker's strategy is a path; namely, the attacker can visit some node u in state $j+1$ only if it is a neighbour of the node v in state j recorded by $\omega_{j,(v,u)}$ in Eqs.(4.25)–(4.26). Eqs.(4.27)–(4.28) initialize the time and update it at the following states. Finally, with Eqs.(4.29)–(4.31), z_S captures whether the attacker is caught by S.

Now we propose ***betterAO*** to efficiently achieve a better response. This algorithm is similar to Dijkstra's algorithm (see Algorithm 7). Let's say the *length* of an attacker path is the probability of being caught by the given $\mathbf{x}$. Variable $\Delta x[v]$ maintains, for all $v \in V$, the shortest so-far-known acyclic path from v_0^a to v. In the main loop, the node with minimum Δx is extracted from Q (Line 4). Then in Lines 8–14, Δx of each neighbor of v is updated with $\Delta x[v]$, where in Line 9 a best time point t is calculated such that reaching u at t minimizes the probability of attacker being caught ($\phi_u(S,t) = 1$ if in S at least one defender resource stays at u at time t, and 0 otherwise). $prev[u]$ maintains the previous node of u in the so-far-known shortest path, with which an attacker path is constructed in the end (Line 15).

4.4 Improving the Scalability

EIGS has improved its scalability by deploying efficient better oracles. However, it cannot efficiently solve the best oracle of large scale games due to the large strategy spaces. In this section, we further improve scalability without significantly sacrificing the defender's utility much. Our approach is motivated by the intuition that the attacker wants to escape as soon as possible and, correspondingly, each police officer goes to one certain node with maximum speed and then waits for the attacker.

4.4.1 A Reduced Game of EIG

$\mathcal{A}$ is reduced to $\underline{\mathcal{A}}$, which satisfies the condition that for every two consecutive states $a_j = (v_j, t_j^a)$ and $a_{j+1} = (v_{j+1}, t_{j+1}^a)$ of each attacker strategy $A \in \underline{\mathcal{A}}$, $t_{j+1}^a = t_j^a + t_{(v_j, v_{j+1})}$. That is, the attacker travels with the maximum speed. $\mathcal{S}$ is reduced to $\underline{\mathcal{S}}$, which satisfies the condition that for each defender strategy $S \in \underline{\mathcal{S}}$, the schedule S^r has the following form: $S^r = \langle (v_0^r, 0, 0), (v_1^r, dist(v_0^r, v_1^r), t_{max}) \rangle$, where $v_1^r \in V \setminus \{v_\infty\}$. That is, the defender goes to some node with maximum speed and waits there. Obviously, $\underline{\mathcal{A}} \subseteq \mathcal{A}$ and $\underline{\mathcal{S}} \subseteq \mathcal{S}$. Now we have the following property about the relation of $\underline{\mathcal{A}}$ and $\underline{\mathcal{S}}$.

Theorem 4.2. *Given a defender strategy* $\mathbf{x}$ *with its support set* $\mathcal{S}_{\mathbf{x}} \subseteq \underline{\mathcal{S}}$*, there is an attacker optimal strategy* $A^* \in \underline{\mathcal{A}}$.

Proof. Suppose that there is not an optimal strategy $A^* \in \underline{\mathcal{A}}$. This means that if A is an optimal strategy, then $A \in \mathcal{A} \setminus \underline{\mathcal{A}}$ and $U_a(\mathbf{x}, A^*) < U_a(\mathbf{x}, A)$.

In any strategy $S \in \underline{\mathcal{S}}$, the schedule S^r has the following form: $S^r = \langle (v_0^r, 0, 0), (v^r, t^{r,in}, t_{max})\rangle$, where $0 \leq t^{r,in} = dist(v_0^r, v_1^r) \leq t_{max}$. Suppose the optimal strategy $A = \langle a_1 = (v_0^a, t_0^a = 0), \ldots, a_j = (v_j, t_j^a), \ldots, a_{k-1} = (v_{k-1}, t_{k-1}^a), a_k = (v_\infty, t_k^a)\rangle$. Keeping the nodes' sequence order of A , we can obtain $\underline{\mathcal{A}}$'s strategy $A^* = \langle a_1^* = (v_0^a, t_0^{a^*} = 0), \ldots, a_j^* = (v_j, t_j^{a^*}), \ldots, a_{k-1}^* = (v_{k-1}, t_{k-1}^{a^*}), a_k^* = (v_\infty, t_k^{a^*})\rangle$, where $t_j^{a*} = t_{(v_{j-1}, v_j)} + t_{j-1}^{a*}(j > 0)$. Then $t_j^{a^*} \leq t_j^a$ $(\forall a_j^* \in A^*, a_j \in A)$ as $t_j^a \geq t_{(v_{j-1}, v_j)} + t_{j-1}^a(j > 0)$.

Moreover, we define $z^*_{S^r, a_j^*} \in \{0, 1\}$ and $z_{S^r, a_j} \in \{0, 1\}$ $(a_j^* \in A^*, a_j \in A)$ as: $z^*_{S^r, a_j^*} = 1$ if $v^r = v_j, t^{r,in} \leq t_j^{a*}$, otherwise $z^*_{S^r, a_j^*} = 0$; and $z_{S^r, a_j} = 1$ if $v^r = v_j, t^{r,in} \leq t_j^a$, otherwise $z_{S^r, a_j} = 0$. Then, $z^*_{S^r, a_j^*} \leq z_{S^r, a_j}$ as $t_j^{a^*} \leq t_j^a$ $(\forall a_j^* \in A^*, a_j \in A)$.

Furthermore, we define $z_S^* \in \{0, 1\}$ and $z_S \in \{0, 1\}$ $(S \in \mathcal{S}_{\mathbf{x}})$ as: $z_S^* = 1$ if $\exists z^*_{S^r, a_j^*} = 1 (r \in \mathcal{R}, a_j^* \in A^*)$, otherwise $z_S^* = 0$; and $z_S = 1$ if $\exists z_{S^r, a_j} = 1 (r \in \mathcal{R}, a_j \in A)$, otherwise $z_S = 0$. Then, $z_S^* \leq z_S$ as $z^*_{S^r, a_j^*} \leq z_{S^r, a_j}$ $(\forall a_j^* \in A^*, a_j \in A)$.

So the attacker's utility $U_a(\mathbf{x}, A^*) = \sum_{S \in \mathcal{S}_{\mathbf{x}}} (1 - z_S^*) x_S \geq \sum_{S \in \mathcal{S}_{\mathbf{x}}} (1 - z_S) x_S = U_a(\mathbf{x}, A)$, which causes a contradiction. □

Therefore, the utility obtained from the optimal defender strategy of the reduced game $(\underline{\mathcal{S}}, \underline{\mathcal{A}})$ is the same as the one in the semi-reduced game $(\underline{\mathcal{S}}, \mathcal{A})$ by Theorem 4.2. This means that *bestAO* cannot find an improving strategy for the attacker if the defender strategy is an equilibrium of $(\underline{\mathcal{S}}, \underline{\mathcal{A}})$.

Theorem 4.3. *EIGS within* $(\underline{\mathcal{S}}, \underline{\mathcal{A}})$ *converges to an NE.*

Proof. EIGS within $(\underline{\mathcal{S}}, \underline{\mathcal{A}})$ will converge because both $\underline{\mathcal{S}}$ and $\underline{\mathcal{A}}$ are finite. Let $(\mathbf{x}, \mathbf{y})$ be the output of *EIGS* when *EIGS* converges. We know that $(\mathbf{x}, \mathbf{y})$ is an NE in the restricted game. In addition, when *EIGS* converges, both players' response oracles cannot find improving strategy outside of the restricted games. Then we have $U_d(\mathbf{x}, \mathbf{y}) \geq U_d(\mathbf{x}', \mathbf{y})$

```
Algorithm 8: advancedAO(x)
1  $V_{\mathbf{x}} \leftarrow \{v_i^r \mid x_S > 0, (v_i^r, t_i^{r,in}, t_i^{r,out}) \in S^r \in S\}$;
2  $V_f \leftarrow$ DFS$(G, V_{\mathbf{x}}, v_0^a)$, $V_{\mathbf{x},f} \leftarrow V_{\mathbf{x}} \cap V_f$;
3  if $v_\infty \in V_f$ then
4      $\mathcal{A}^* \leftarrow \{P(dist_{(V_f,V_{\mathbf{x},f})}(v_0^a, v_\infty))\}$;
5  else
6      $V_b \leftarrow$ DFS$((V, \{(u,v) \mid (v,u) \in E\}), V_{\mathbf{x}}, v_\infty)$;
7      $V_{\mathbf{x},b} \leftarrow V_{\mathbf{x}} \cap V_b$;
8      $E \leftarrow E \cup \{(v_0^a, v) \mid v \in V_{\mathbf{x},f}\} \cup \{(v, v_\infty) \mid v \in V_{\mathbf{x},b}\}$;
9      $t_{v_0^a,v} \leftarrow dist_{(V_f,V_{\mathbf{x},f})}(v_0^a, v), \forall v \in V_{\mathbf{x},f}$;
10     $t_{v,v_\infty} \leftarrow dist_{(V_b,V_{\mathbf{x},b})}(v, v_\infty), \forall v \in V_{\mathbf{x},b}$;
11     $V_c \leftarrow (V \setminus (V_f \cup V_b)) \cup (V_{\mathbf{x},f} \cup V_{\mathbf{x},b} \cup \{v_0^a, v_\infty\})$;
12     $E_c \leftarrow \{(u,v) \mid u, v \in V_c, (u,v) \in E\}$;
13     $\mathcal{A} \leftarrow \{RedbestAO(\mathbf{x})\}$;
14     if $\mathcal{A} \neq \emptyset$ then
15         $\mathcal{A}^* \leftarrow \{P(dist_{(V_f,V_{\mathbf{x},f})}(v_0^a, v_1)) \cup (v_2^a, t_2^a), \cdots, (v_{l-2}^a, t_{l-2}^a) \cup$
             $P(dist_{(V_b,V_{\mathbf{x},b})}(v_{l-1}^a, v_\infty)) \mid (v_0^a, t_0^a), \cdots, (v_\infty, t_l^a) \in \mathcal{A}\}$;
16 return $\mathcal{A}^*$.
```

($\forall \mathbf{x}'$), and $U_a(\mathbf{x}, \mathbf{y}) \geq U_a(\mathbf{x}, \mathbf{y}')$ ($\forall \mathbf{y}'$) in the full game. Then, $(\mathbf{x}, \mathbf{y})$ is an NE in the full game. $\square$

4.4.2 An Approximate Algorithm of EIG

Based on the reduced game $(\underline{\mathcal{S}}, \underline{\mathcal{A}})$, we can redesign the oracles in Algorithm 5. Here the new best DO, *RedbestDO*, has the same objective function as *bestDO* and it has the following constraints:

$$\textstyle\sum_{v \in V \setminus \{v_\infty\}} s_v^r = 1 \quad \forall r \tag{4.33}$$

$$\textstyle\sum_{r \in \mathcal{R}} s_v^r \leq 1 \quad \forall v \in V \setminus \{v_\infty\} \tag{4.34}$$

$$z_A \leq \textstyle\sum_{j,r,dist(v_0^r, v_j^A) \leq t_j^A} s_{v_j^A}^r \quad \forall A \tag{4.35}$$

$$s_v^r, z_A \in \{0, 1\} \tag{4.36}$$

$s_v^r = 1$ indicates that resource r is deployed to node v. Eq.(4.33) ensures that each resource will be assigned to one node, which is protected by at most one resource (Eq.(4.34)). z_A indicates whether the attacker's path is intercepted by the defender. $z_A = 1$

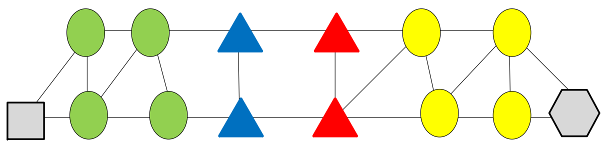


FIGURE 4.2: A scenario: the defender will only randomly appear at the triangle nodes to interdict the attacker but will not appear at other nodes; now the attacker needs to find an optimal strategy considering travel time from his current position (the square node on the left) to the external world (the hexagon node on the right).

only when there is one node on the path where there is one resource reaching there before the attacker (reflected by $dist(v_0^r, v_j^A) \leq t_j^A$) enforced by Eq.(4.35).

The new better DO, *RedbetterDO*, is similar to *betterDO*, except that *RedbetterDO* only deploys each resource to one node. The new best AO, *RedbestAO*, is obtained from *bestAO* by changing the method for updating time in Eq.(4.28) to $t_{j+1}^a = t_j^a + \sum_{(v,u)\in E} t_{(v,u)}\omega_{j,(v,u)}$ to reflect the fact that the attacker will travel with maximum speed. Similarly, the new better AO, *RedbetterAO*, is obtained from *betterAO* by modifying Line 9 in Algorithm 7 to: if $\exists S^r s.t. z_{s_2^r,(u,t^a[v]+t_{v,u})} = 1, \forall S \in \overline{\mathcal{S}}_v$, then $\Phi(t) \leftarrow \Phi(t) + x_S$.

In the above new oracles, *RedbestAO* still has to add $|V|$ variables (i.e., $A_{j,v}$) for each state in A, which will dramatically slow it down. To speed it up, we further reduce the attacker strategy space by graph contraction.

***advancedAO*:** The advanced AO is based on the following intuition: for the nodes where no defender will appear, the attacker can travel freely through them without being caught at any time; but, for the node where the defender will appear, he has to find an optimal strategy, e.g., the optimal time, to travel through these nodes to avoid being caught. For example, as shown in Figure 4.2, to reach the external world, the attacker needs to travel through the circle and triangle nodes. The attacker knows that he will not be caught at the circle nodes but will probably be caught at the triangle nodes. So he can freely take any strategy (path) to travel through the circle nodes but wants to take an optimal strategy (path) to travel through the triangle nodes. To reflect these facts in a simplified network, the circle nodes and their adjacent edges can be deleted. Then four new edges from the square node to each blue triangle node and from each red triangle node to the hexagon node will be added with the travel time of the corresponding special shortest path[1] in the original network.

This contraction procedure is shown in Algorithm 8. $V_{\mathbf{x}}$ stores nodes in the support set of $\mathbf{x}$ (Line 1). V_f, a set of nodes that can be reached from v_0^a without traversing the nodes in $V_{\mathbf{x}}$, is given by DFS$(G, V_{\mathbf{x}}, v_0^a)$ (Line 2), where DFS$(G, V_{\mathbf{x}}, v_0^a)$ is obtained from the classic Depth First Search algorithm by ending each path at the nodes in $V_{\mathbf{x}}$. That is, a node v is pushed into the stack only if v is unvisited and $v \notin V_{\mathbf{x}}$. If $v_\infty \in V_f$, then $P(dist_{(V_f,V_{\mathbf{x},f})}(v_0^a, v_\infty))$ is an optimal strategy that will not be intercepted by the defender (Line 4). Here $P(dist_{(V_f,V_{\mathbf{x},f})}(v_0^a, v_\infty))$ is the shortest path from v_0^a to v_∞ on the network $(V_f, \{(u, v) \mid (u, v) \in E, u \in V_f \setminus V_{\mathbf{x},f}, v \in V_f\})$ with corresponding travel time $dist_{(V_f,V_{\mathbf{x},f})}$ (v_0^a, v_∞). Otherwise, the network G is contracted (Lines 6–12) and *RedbestAO*$(\mathbf{x})$ is called for the contracted network (Line 13). V_b stores nodes reached from v_∞ in the backward network of G (Line 6). After that, new edges are added (Line 8) with corresponding travel time (Lines 9 and 10), and then, the contracted network (V_c, E_c) is generated in Lines 11 and 12. Finally, Line 15 maps the generated strategy (Line 13) to the original network.

Now we can obtain ***RedAD***, an efficient algorithm based on the reduced game of EIG, from Algorithm 5 by replacing *bestDO*, *betterDO*, *bestAO*, and *betterAO* with *RedbestDO*, *RedbetterDO*, *advancedAO*[2], and *RedbetterAO*, respectively.

4.5 Experimental Evaluation

We evaluate our model and algorithms with numerical experiments. All LPs and MILPs are solved with CPLEX (version 12.6). All computations are performed on a machine with a 3.20GHz quad core CPU and 16.00GB memory. All points in the figures are averaged over 30 samples. We use the Grid model with Random Edges (GRE) to generate

[1]They are not necessary the shortest paths in the network because they do not contain the triangle nodes, except for the start or end node.

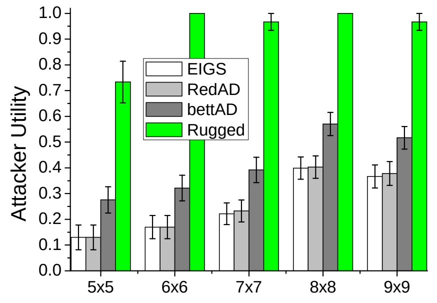
[2]*advancedAO* can be applied to the full game $(\mathcal{S}, \mathcal{A})$ by replacing *RedbestAO* with *bestAO* in Line 13 of Algorithm 8. Actually, we also propose a two-stage approach for the full game $(\mathcal{S}, \mathcal{A})$ which uses *RedAD* to initialize $(\mathcal{S}', \mathcal{A}')$ in Line 1 of Algorithm 5 to obtain algorithm *EIGS-TS*, and then obtain a heuristic algorithm *EIGS-TS-H* where *bestAO* is replaced by *advancedAO*. However, they cannot improve much in solution quality and scalability.

urban road network topologies [71]. GRE is a planar connected graph made of a $L \times W$ square grid of nodes, where horizontal/vertical edges between neighbors are controlled by probability p, and diagonal ones by q. The values of p spread in $[0.3, 0.9]$ and q in $[0.1, 0.7]$ to best match realistic road networks. We add a node v_∞ that is connected to all the nodes at the border of the area to generate the entry and exit links.

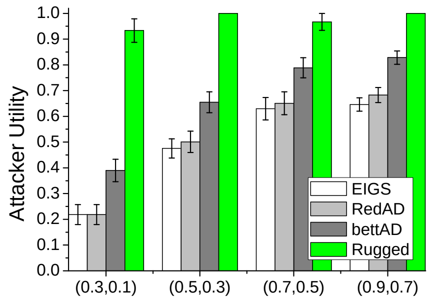
In the traffic model, $C_e = 6$, and T_e is randomly chosen in [1,10]. The traffic demand of each entry link is randomly chosen in [0,6], and the turning preference matrix R is uniformly generated. In EIG, v_0^a is fixed to the center node of the area, and all the m defender resources are initially uniformly distributed on the network. The exit nodes D are randomly chosen from the nodes at the border of the area. The defender travels faster than the attacker because the police can circumvent traffic constraints unless otherwise specified. Central parameter values are chosen as $L \times W = 8 \times 8$ nodes, $(p, q) = (0.4, 0.2)$, $|D| = 10$ exit nodes and $m = 4$ resources unless otherwise specified.

4.5.1 Solution Quality

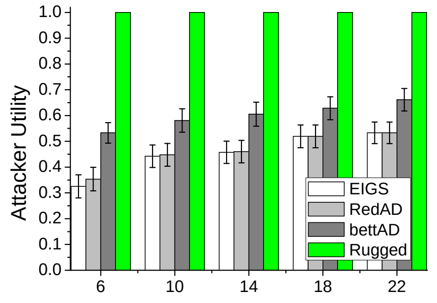
We compare the solution quality of our approaches with two baselines. The first baseline is *Rugged* [20], where the defender resources are deployed to intercept the attacker's path without considering the travel time on paths. *Rugged* is a representative solution in the urban network security domain for the following reasons: 1) it is an extension of the min-cut method that the defender resources are uniformly distributed on the $v_0^a - v_\infty$ min-cut [70] without the travel time constraint; 2) it has the same solution quality as its variation of *Snares* [25] because *Snares* deploys *Rugged*'s best response oracles, i.e., *Snares* is just an improvement of *Rugged* in scalability; 3) its performance in solution quality is better than *MiCANS* [21], which is an approximate algorithm for urban network security based on multiple cuts. The second baseline is the approximate algorithm *bettAD* where *EIGS* does not call the best response oracles. Comparing *bettAD* with *RedAD* in solution quality will further show the importance of developing *RedAD*. The solution quality of strategy $\mathbf{x}$ in *Rugged*, *RedAD*, and *bettAD* is assessed in terms of $U_a(\mathbf{x}, A)$, where A is obtained by using *bestAO*. A lower attacker utility indicates a higher defender utility given the zero-sum assumption.



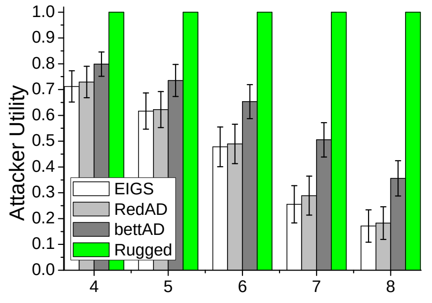
(a) Attacker utility: $L \times W$



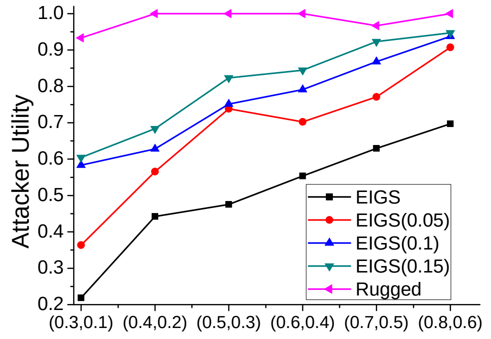
(b) Attacker utility: (p, q)



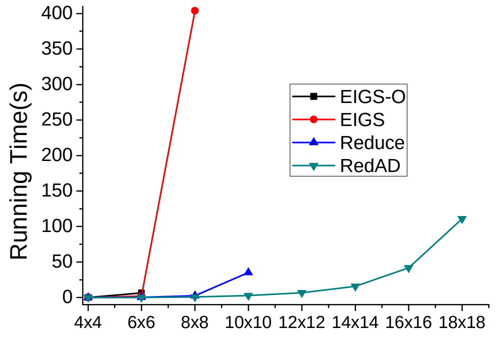
(c) Attacker utility: $|D|$



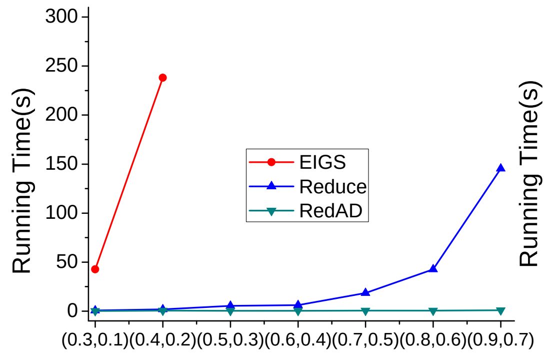
(d) Attacker utility: m



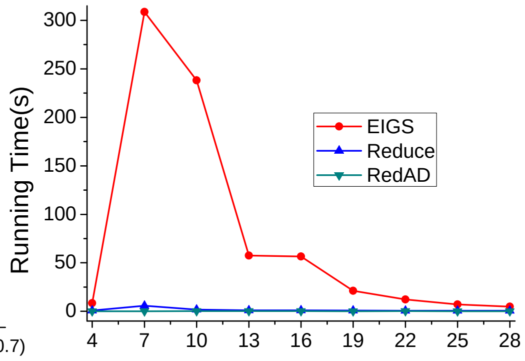
(e) Robustness: (p, q)



(f) Running time: $L \times W$



(g) Running time: (p, q)



(h) Running time: $|D|$

FIGURE 4.3: Solution quality: (a)–(d); Robustness: (e); Scalability: (f)–(h).

Figures 4.3a–4.3d compare the solution quality obtained from our approaches with those obtained from baselines varying: the number of intersections, denoted by $L \times W$, with $|D| = L$ in Figure 4.3a; edge probabilities in Figure 4.3b; the number of exit nodes in Figure 4.3c; and the number of resources in Figure 4.3d. Results show that *Rugged* performs the worst in all cases because it is designed without considering players' travel time on roads. Meanwhile, *EIGS* performs the best. *bettAD* performs significantly worse than *EIGS*. However, *RedAD* achieves good solution quality, which has similar performance to *EIGS* and significantly outperforms *bettAD* in most cases.

In terms of robustness, the defender may face execution uncertainty caused by the unpredictable delays associated with congestion, traffic signals, etc. To analyze the performance of *EIGS* in the presence of uncertainty, we add noise θ to the travel time. So the actual travel time $\hat{t_e} = t_e \times (1+\theta)$. Figure 4.3e shows the attacker utility of *EIGS* under different degrees of uncertainty. Here we can see that while the performance of *EIGS* decreases as θ is increased, it still outperforms *Rugged*.

4.5.2 Scalability

We compare the scalability of *EIGS* with a baseline, *EIGS-O*, which runs *EIGS* under the double oracle framework adopted by [25, 26]. This shows the effectiveness of our new double oracle framework. In addition, we evaluate the efficiency of *RedAD* with another baseline, *Reduce*, where *RedAD* calls *RedbestAO* instead of *advancedAO*.[1]

Figures 4.3f–4.3h present the runtime in seconds varying different variables similar to the setting of solution quality. From Figure 4.3f, we can see that *EIGS-O* does not scale up to the network with 8×8 intersections, so we do not show the result of *EIGS-O* in Figures 4.3g and 4.3h. Results show that: 1) Our new double oracle framework outperforms the old one by comparing *EIGS* with *EIGS-O*; 2) *RedAD* outperforms *EIGS* and *Reduce* significantly; 3) The running time generally increases with the size of the network and the edge probabilities, but it reflects the easy-hard-easy pattern in security games [79]; 4) *RedAD* can scale up to realistic-sized problems, i.e., an urban road network with 324 intersections. Note that in real world examples, police forces typically focus on patrolling their own wards, which typically have the size of up to hundreds of intersections[2]. Thus, our algorithm can scale up to problems with realistic size.

[1]We do not show the running time of *Snares* because it has the same solution quality as *Rugged*, which performs significantly worse than our algorithm in solution quality.

[2]E.g., Singapore Police Force has six divisions, and each division has up to dozens of neighbourhood police posts or police centers distributed in the city.

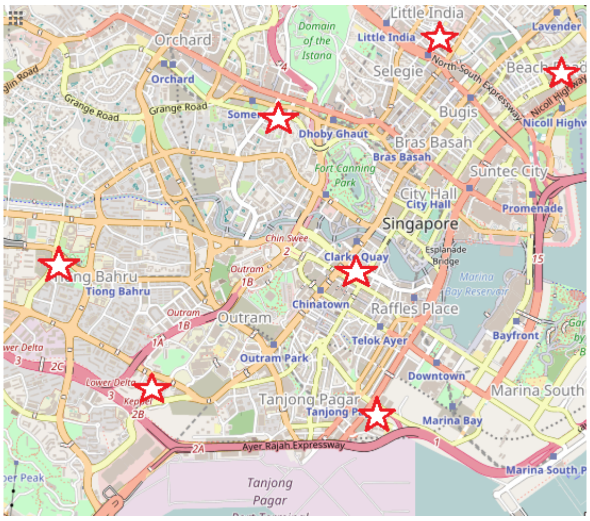


FIGURE 4.4: The real road network

Application on the Real Network: We test the performance of RedAD on the real-world road network with actual traffic data from Land Transport Authority (LTA) of Singapore. We extract the information about the road network in the area between longitudes 103.8181 and 103.8657 and latitudes 1.2668 and 1.3081 as shown in Figure 4.4 from OSM[1]. In this dataset, there are 371 nodes with 29 exit nodes on the border and 678 links whose travel time is transferred from the speed data. The positions of seven police stations are labelled by red pentagrams in Figure 4.4. A total of 30 cases with different attacker initial positions are tested. For each case, *RedAD* takes only 14.7 seconds to deploy seven resources.

4.6 Chapter Summary

We present a novel game-theoretic model for interdicting the escaping attacker on urban traffic networks. We show that finding a CTME is NP-hard. Therefore, we propose an efficient double oracle framework, *EIGS*, based on novel MILPs for best response oracles and heuristic algorithms for better response oracles. To solve large-scale problems, we develop *RedAD* to significantly reduce the strategy space for both players. Experimental results show that our algorithms obtain a robust solution that significantly outperforms baselines and can scale up to realistic-sized problems.

[1]www.openstreetmap.org

Chapter 5

Computing Team-Maxmin Equilibria in Zero-Sum Multiplayer Extensive-Form Games

This chapter[1] focuses on computing TMEs [15] in zero-sum multiplayer Extensive-Form Games (EFGs). We first study the inefficiency caused by computing a CTME (i.e., TMECor or TMECom) and then transforming it into the mixed strategy profile of the team and theoretically show that this inefficiency can be arbitrarily large in E-FGs. Second, to efficiently solve the non-convex program for finding TMEs directly, we develop the Associated Recursive Asynchronous Multiparametric Disaggregation Technique (ARAMDT) to approximate multilinear terms in the program by using a digit-wise discretization of one variable in the term and then transform the program into a mixed-integer linear program (MILP) with two novel techniques: 1) an asynchronous precision method, where we can use different precision levels to approximate these terms even they include the same discretized variable to reduce the number of constraints and variables for approximation; and 2) an associated constraint method, where we exploit the relation between these terms to reduce the feasible solution space of MILP in the ARAMDT. Third, we develop a novel algorithm to efficiently compute

[1]The work in this chapter has been published as **Youzhi Zhang** and Bo An. Computing team-maxmin equilibria in zero-sum multiplayer extensive-form games. *Proceedings of the 34th AAAI Conference on Artificial Intelligence* (**AAAI**), accepted, 2020.

TMEs within any given accuracy based on the ARAMDT, which uses novel techniques to iteratively increase the precision levels for part of non-linear terms. Finally, we e-valuate our algorithm by experiments showing that our algorithm is dramatically faster than baselines.

5.1 The Inefficiency of Correlated Strategies

It is very difficult to solve the non-convex Problem (2.6) [15] even using the state-of-the-art global optimization solver BARON [27]. Instead of solving Problem (2.6) to compute a TME, one potential approach is to compute a CTME (i.e., TMECor or TMECom) first and then transform it into the team's Mixed Strategy Profile (Transformed-MSP(TMSP)), as done in normal-form games [14]. This section shows that this approach can cause an arbitrarily large loss.

To measure the inefficiency of this TMSP, we define the Price of Correlated strategies (PoC)[1]. That is the inefficiency caused by that team players use this TMSP to independently play against the adversary. Formally, PoC $= \frac{v_m}{v_c}$, where v_m is the TME value and v_c is the team's utility obtained from the TMSP. The best algorithm [14], to the best of our knowledge, to obtain a TMSP is: Given a CTME strategy for the team: $x \in \Delta(\Pi_T)$ (the probability distribution over the set of joint normal-formal strategies), player 1's mixed strategy $x_1(\pi_1) = \sum_{\pi' \in \Pi_{T\setminus\{1\}}} x(\pi_1, \pi')(\forall \pi_1 \in \Pi_1)$; player j's mixed strategy $x_j(\pi_j) = \frac{1}{|\text{sup}_j|}$ if $|\text{sup}_j| > 0$, otherwise $x_j(\pi_j) = 0$ $(\forall j \in T \setminus \{1\}, \pi_j \in \Pi_j)$ where $\text{sup}_j = \{\pi_j \mid \pi_j \in \Pi_j, \exists \pi_T, \pi_j \in \pi_T, x(\pi_T) > 0\}$, i.e., the set of strategies player j plays with nonzero probability in x. This algorithm returns the best strategy for the team among the strategies obtained by exchanging each player of the team with player 1. From previous experiments [14], we can see the large gap between v_m and v_c in normal-form games. Now we theoretically show that PoC can be arbitrarily large.

Proposition 5.1. *PoC can be arbitrarily large in EFGs.*

[1]PoC is different from the price of uncorrelation ($\frac{v'_c}{v_m}$) [14], i.e., the inefficiency caused by that team players do not correlate their strategies when they can (and then obtain utility v'_c).

Proof. Consider an EFG with n players ($n-1$ teammates) and $m(>2)$ actions per player's decision node where each level of the game tree forms a unique information set belonging to one player (see Figure 5.1 for a game tree with $n=3$), and the payoffs of the team at terminal nodes ($\pi_i = a(1 \le a \le m)$ is a pure normal-form strategy) are:

$$U_T(\pi_1,\ldots,\pi_n)=\begin{cases}1 & \text{if } \pi_1=\pi_2=\cdots=\pi_n\\ \dfrac{1}{m+1} & \text{if } \pi_1=1, \pi_i=2(\forall i\in T\backslash\{1\})\\ -\dfrac{m+2}{m+1} & \text{if } \pi_1=2, \pi_i=1(\forall i\in T\backslash\{1\})\\ 0 & \text{otherwise}\end{cases}$$

To obtain a TMSP, we need to find a CTME first. A CTME (the TMECom and TMECor are the same in this case) is the strategy profile prescribing that the team plays each joint strategy $\pi_1 = \cdots = \pi_{n-1} = a(1 \le a \le m)$ with probability $\frac{1}{m}$.[1] Now, by the aforementioned transformed algorithm [14], the corresponding (unique) TMSP prescribes that player $i(\in T)$ plays $\pi_i = a(1 \le a \le m)$ with probability $\frac{1}{m}$, while player n is indifferent between playing any pure strategies given this TMSP. Then, $v_c = 0$.

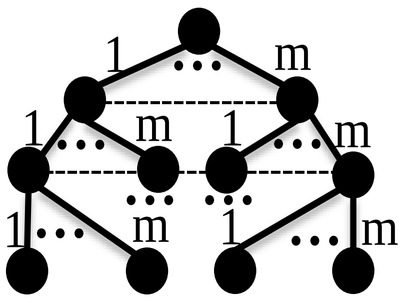


FIGURE 5.1: Example

Even though it is difficult to determine a TME analytically, we can find a good lower bound for the TME value through an NE because a TME is an NE maximizing the team's utility. This equilibrium strategy $(1, 2, \ldots, 2, 3)$ prescribes that player 1 plays $\pi_1 = 1$, player $i(2 \le i \le n-1)$ plays $\pi_i = 2$, while player n plays $\pi_n = 3$. The team's utility is $U_T(1, 2, \ldots, 2, 3) = \frac{1}{m+1}$. Basically, any team member will obtain 0 from unilateral deviation while the adversary is indifferent between playing any pure strategies.

[1]First, the team, obtaining utility $\frac{1}{m}$ in this profile, cannot improve its utility by playing other joint strategies with nonzero probability because all other joint strategies are weakly dominated as they provide a utility less than $\frac{1}{m}$ to the team for every adversary strategy. Second, the team must play each joint strategy of the form $\pi_1 = \cdots = \pi_{n-1}$ with nonzero probability because the team will receive utility 0 if any such a form with a' is played with probability 0, but the adversary n plays a'. Therefore, each such a form is played with probability $\frac{1}{m}$ by symmetry.

Therefore, PoC $= \frac{v_m}{v_c} \geq \frac{\frac{1}{m+1}}{0} = \infty$. □

5.2 Associated Recursive Asynchronous MDT

This section proposes novel techniques to directly solve the non-convex Problem (2.6) to compute TMEs. Specifically, we develop the Associated Recursive Asynchronous Multiparametric Disaggregation Technique (ARAMDT) to approximate multilinear terms in Eq.(2.6b). The ARAMDT is developed based on the MDT [40, 42], which uses a digit-wise discretization of one variable in the bilinear term for approximation and then transforms the bilinear program into a mixed-integer linear program (MILP). The ARAMDT has three additional important features: 1) the Asynchronous MDT (AMDT) (i.e., adding the asynchronous precision method to the MDT) to reduce the number of constraints and new variables for approximating bilinear terms; 2) the Recursive AMDT (RAMDT) (i.e., recursively deploying the AMDT) to approximate multilinear terms which cannot be directly approximated by the AMDT; and 3) the Associated RAMDT (ARAMDT) (i.e., adding associated constraints to the RAMDT) to reduce the feasible solution space of MILP in the RAMDT.

5.2.1 Asynchronous MDT

To transform the bilinear program into MILP while reducing the number of constraints and new variables for approximating bilinear terms, we develop the AMDT which can use different precision levels to approximate bilinear terms even they include the same discretized variable. The idea is that: If we do not need high precision levels to approximate some bilinear terms, the AMDT can just use low precision levels to do that, which reduces the number of constraints and new variables because this number of constraints and variables increases with the precision levels. Given a bilinear term $w = y_i y_j$ ($y_i, y_j \in [0, 1]$), we approximate y_i by using a binary number with powers (precision levels) $\mathbb{Z}_{y_i} = \{-1, \ldots, -Z_{y_i}\}$:

$$y_i = \textstyle\sum_{z \in \mathbb{Z}_{y_i}} 2^z \lambda_{i,z} + \Delta y_i \tag{5.1}$$

where $\lambda_{i,z} \in \{0,1\}$ and $\Delta y_i \in [0, 2^{-Z_{y_i}}]$ is the slack variable. After that, we approximate w by using powers $\mathbb{Z}_w = \{-1, \dots, -Z_w\}$ with $Z_w \le Z_{y_i}$:

$$w = \textstyle\sum_{z \in \mathbb{Z}_w} 2^z y_{i,j,z} + \Delta w \tag{5.2}$$

where $y_{i,j,z} = \lambda_{i,z} y_j$ can be represented by:

$$0 \le y_{i,j,z} \le \lambda_{i,z} \quad \forall z \in \mathbb{Z}_w \tag{5.3a}$$

$$0 \le y_j - y_{i,j,z} \le 1 - \lambda_{i,z} \quad \forall z \in \mathbb{Z}_w \tag{5.3b}$$

$\Delta w = (y_i - \sum_{z \in \mathbb{Z}_w} 2^z \lambda_{i,z}) y_j$ $(0 \le y_i - \sum_{z \in \mathbb{Z}_w} 2^z \lambda_{i,z} = \Delta y_i + \sum_{-Z_{y_i} \le z < -Z_w} 2^z \lambda_{i,z} \le 2^{-Z_w})$ cannot be represented exactly by linear constraints and then is approximated by using the McCormick relaxation [45]:

$$0 \le \Delta w \le y_i - \textstyle\sum_{z \in \mathbb{Z}_w} 2^z \lambda_{i,z} \tag{5.4a}$$

$$2^{-Z_w}(y_j - 1) + y_i - \textstyle\sum_{z \in \mathbb{Z}_w} 2^z \lambda_{i,z} \le \Delta w \le 2^{-Z_w} y_j \tag{5.4b}$$

Here, the AMDT replaces Δy_i in the standard MDT [40] with $y_i - \sum_{z \in \mathbb{Z}_w} 2^z \lambda_{i,z}$ for representing Δw because $Z_w \le Z_{y_i}$. Therefore, the AMDT can use different powers to approximate bilinear terms even they include the same discretized variable, which will reduce the number of constraints if we do not need to use more powers to approximate some bilinear terms, e.g., when w is already equal to $y_i y_j$.

5.2.2 Recursive Asynchronous MDT

EFGs with $n > 3$ will result in multilinear terms in Eq.(2.6b), where the AMDT cannot be applied directly. Therefore, we develop the RAMDT, which recursively uses a new variable to replace each bilinear part of the multilinear term in Eq.(2.6b) until the multilinear term is replaced by a variable (representing a bilinear term), and then the bilinear term is approximated by the AMDT. More specifically, for each $\sigma_T \in \Sigma'_T (\subseteq \Sigma_T)$

reaching a terminal node with the multilinear term $\prod_{i\in T} r_i(\sigma_T(i))$ in Eq.(2.6b), we recursively define a set of bilinear terms $BL_{\sigma_T} = \{w_1(\sigma_{T_1}), \ldots, w_{n-2}(\sigma_{T_{n-2}})\}$ such that:

$$w_i(\sigma_{T_i}) = r_i(\sigma_{T_i}(i))w_{i+1}(\sigma_{T_{i+1}}) = \prod_{i\le j\le n-1} r_j(\sigma_{T_i}(j))$$

where $w_i(\sigma_{T_i}) \in [0,1](1 \le i \le n-2)$, σ_{T_i} represents the joint sequence of players $i,\ldots,n-1$ as part of σ_T with $\sigma_{T_1} = \sigma_T$, $\sigma_{T_i}(j)$(i.e., $\sigma_T(j)$) is the sequence of player j in σ_{T_i}, and $w_{n-1}(\sigma_{T_{n-1}}) = r_{n-1}(\sigma_{T_{n-1}}(n-1))$. Therefore, each multilinear term can be represented by a bilinear term. To be consistent, we discretize the variable on the left hand side of each bilinear term in BL_{σ_T}.

Denote $BL = \bigcup_{\sigma_T\in\Sigma'_T} BL_{\sigma_T}$. By using $w_1(\sigma_T)$ to replace $\prod_{i\in T} r_i(\sigma_T(i))$ in Eq.(2.6b) through the RAMDT, an upper bound of Problem (2.6) is computed by:

$$\max_{r_1,\ldots,r_{n-1}} v(\mathcal{I}_n(\emptyset))_{RAMDT} \tag{5.5a}$$

$$v(\mathcal{I}_n(\sigma_n)) - \sum_{I_{n,j}\in I_n:\text{seq}_n(I_{n,j})=\sigma_n} v(I_{n,j})$$
$$\le \sum_{\sigma_T\in\Sigma_T} U_T(\sigma_T,\sigma_n)w_1(\sigma_T) \quad \forall\sigma_n\in\Sigma_n \tag{5.5b}$$

$$\text{Eqs.}(2.5a)-(2.5c) \quad \forall i\in T \tag{5.5c}$$

$$\text{Eq.}(5.1) \quad \forall r_i(\sigma_T(i)), i\in T\backslash\{n-1\}, \sigma_T\in\Sigma'_T \tag{5.5d}$$

$$\text{Eqs.}(5.2)-(5.4b) \quad \forall w_i(\sigma_{T_i})\in BL \tag{5.5e}$$

Theorem 5.1. *The solution of Problem (5.5) yields an upper bound for Problem (2.6), i.e.,* $v(\mathcal{I}_n(\emptyset))_{RAMDT} \ge v(\mathcal{I}_n(\emptyset))$.

Proof. The variable $w_i(\sigma_{T_i})$ in Problem (5.5) may not be exactly equal to $r_i(\sigma_{T_i}(i))w_{i+1}(\sigma_{T_{i+1}})$ due to the power $Z_{w_i(\sigma_{T_i})}$. In fact, by Eqs.(5.1)–(5.4b), $|w_i(\sigma_{T_i}) - r_i(\sigma_{T_i}(i))w_{i+1}(\sigma_{T_{i+1}})| \le 2^{-Z_w}w_{i+1}(\sigma_{T_{i+1}})$. It means that, some assignments for $w_i(\sigma_{T_i})$, $r_i(\sigma_{T_i}(i))$, and $w_{i+1}(\sigma_{T_{i+1}})$ without satisfying $w_i(\sigma_{T_i}) = r_i(\sigma_{T_i}(i))w_{i+1}(\sigma_{T_{i+1}})$ are feasible in Problem (5.5). Thus, Problem (5.5) is a relaxation of Problem (2.6). Therefore, the solution of Problem (5.5) yields an upper bound for Problem (2.6), i.e., $v(\mathcal{I}_n(\emptyset))_{RAMDT} \ge v(\mathcal{I}_n(\emptyset))$. □

Theorem 5.2. *As $Z_{w_i(\sigma_{T_i})}$ $(\forall w_i(\sigma_{T_i}) \in BL)$ in Problem (5.5) approaches ∞, $v(\mathcal{I}_n(\emptyset))_{RAMDT}$ approaches $v(\mathcal{I}_n(\emptyset))$.*

Proof. As $Z_{w_i(\sigma_{T_i})}$ approaches ∞, we have: $\lim_{Z_{w_i(\sigma_{T_i})}\to\infty} |w_i(\sigma_{T_i}) - r_i(\sigma_{T_i}(i))w_{i+1}(\sigma_{T_{i+1}})| \leq \lim_{Z_{w_i(\sigma_{T_i})}\to\infty} 2^{-Z_{w_i(\sigma_{T_i})}} w_{i+1}(\sigma_{T_{i+1}}) = 0$, which means that $w_i(\sigma_{T_i})$ approaches $r_i(\sigma_{T_i}(i))w_{i+1}(\sigma_{T_{i+1}})$. Then, $v(\mathcal{I}_n(\emptyset))_{RAMDT}$ approaches $v(\mathcal{I}_n(\emptyset))$. □

Then, $v(\mathcal{I}_n(\emptyset))$ is reached if $Z_{w_i(\sigma_{T_i})}$ is large enough.

5.2.3 Associated Recursive Asynchronous MDT

By Theorem 5.1, the solution of Problem (5.5) is an upper bound for Problem (2.6) because $w_i(\sigma_{T_i})$ may not be equal to $r_i(\sigma_{T_i}(i))w_{i+1}(\sigma_{T_{i+1}})$, which also makes the feasible solution space very large and then makes it difficult to solve Problem (5.5). Here we aim to reduce the feasible solution space in Problem (5.5) and then solve it efficiently. To do that, we develop the ARAMDT, which adds associated constraints to the RAMDT by exploiting the relation between bilinear terms.

The relation between bilinear terms is based on the constraints for the realization plan in Eqs.(2.5a)-(2.5c). Basically, for all sequences with the last action in the same information set, the sum of probabilities to play these sequences is equal to the probability to play the sequence reaching that information set. For example, in Example 2.1, for bilinear terms $w_1(\sigma_1, \textit{J:/c:c})$, $w_1(\sigma_1, \textit{J:/ccrc:c})$, and $w_1(\sigma_1, \textit{J:/ccrc:f})$, we have $w_1(\sigma_1, \textit{J:/c:c}) = w_1(\sigma_1, \textit{J:/ccrc:c}) + w_1(\sigma_1, \textit{J:/ccrc:f})$. In addition, given $w_1(\sigma_1, \textit{J:/c:r})$, we have $w_1(\sigma_1, \textit{J:/c:c}) + w_1(\sigma_1, \textit{J:/c:r}) = r_1(\sigma_1)r_2(\emptyset) = r_1(\sigma_1)$. That is, we look for the equivalence relation between bilinear terms until these terms are connected to the played probability of a sequence, which is represented by associated constraints. After adding these constraints, the solutions which cannot satisfy these constraints will be ruled out immediately, and then the feasible solution space in Problem (5.5) is smaller.

Algorithm 9 generates associated constraints for the equivalence relation between bilinear terms. This relation is based on the realization plan, which is also based

Algorithm 9: Generate associated constraints

```
1  for $w_i(\sigma_{T_i}) \in BL$, and $j \leftarrow i, \dots, n-1$ do
2      $BL_{\sigma_{T_i\setminus\{j\}}, \mathcal{I}_j(\sigma_{T_i}(j))} \leftarrow BL_{\sigma_{T_i\setminus\{j\}}, \mathcal{I}_j(\sigma_{T_i}(j))} \cup \{w_i(\sigma_{T_i})\}$;
3  isContinue $\leftarrow$ true, $BL' \leftarrow BL$;
4  while isContinue = true do
5      isContinue $\leftarrow$ false;
6      for $i \leftarrow n-2, \dots, 1$; $j \in \{i, \dots, n-1\}$; each $BL_{\sigma_{T_i\setminus\{j\}}, \mathcal{I}_j(\sigma_{T_i}(j))}$ do
7          $w_i(\sigma'_{T_i}) \leftarrow \varnothing, \sigma_j \leftarrow \sigma_{T_i}(j)$;
8          if $i = j \wedge |BL_{\sigma_{T_i\setminus\{j\}}, \mathcal{I}_j(\sigma_j)}| = |\chi(\mathcal{I}_j(\sigma_j))|$ then
9              $w_i(\sigma'_{T_i}) \leftarrow w_i(seq_j(\mathcal{I}_j(\sigma_j)), \sigma_{T_i\setminus\{j\}})$;
10         if $i \neq j \wedge BL_{\sigma_{T_i\setminus\{i\}}}$ in $RM$ then
11             $w_i(\sigma'_{T_i}) \leftarrow r_i(\sigma_i) RM[BL_{\sigma_{T_i\setminus\{i\}}}]$;
12         if $w_i(\sigma'_{T_i}) \neq \varnothing$ then
13             isContinue $\leftarrow$ true;
14             if $w_i(\sigma'_{T_i}) \notin BL'$ then
15                 Line 2, $j \leftarrow i, \dots, n-1$;
16                 $BL' \leftarrow BL' \cup \{w_i(\sigma'_{T_i})\}$;
17                 if $r_i(\sigma'_{T_i}(i)) = 1$ then
18                     Add $w_i(\sigma'_{T_i}) = w_{i+1}(\sigma'_{T_i\setminus\{i\}})$;
19                 if $w_{i+1}(\sigma'_{T_i\setminus\{i\}}) = 1$ then
20                     Add $w_i(\sigma'_{T_i}) = r_i(\sigma'_{T_i}(i))$;
21                 if $r_i(\sigma'_{T_i}(i)) = w_{i+1}(\sigma'_{T_i\setminus\{i\}}) = 1$ then
22                     Add $w_i(\sigma'_{T_i}) = 1$;
23             Add $w_i(\sigma'_{T_i}) = \sum_{w'_i \in BL_{\sigma_{T_i\setminus\{j\}}, \mathcal{I}_j(\sigma_{T_i}(j))}} w'_i$;
24             $RM[BL_{\sigma_{T_i\setminus\{j\}}, \mathcal{I}_j(\sigma_{T_i}(j))}] \leftarrow w_i(\sigma'_{T_i})$;
25         Remove $BL_{\sigma_{T_i\setminus\{j\}}, \mathcal{I}_j(\sigma_j)}$;
```

on information sets of the game tree (see Eq.(2.5b)). Therefore, for a set of bilinear terms, to check whether one associated constraint of a player based on Eq.(2.5b) can be created, we can fix other players' sequences, and then check whether this player has the sequences in this set of bilinear terms involving all actions in the related information set. To do that, we create subsets of bilinear terms $BL_{\sigma_{T_i\setminus\{j\}}, \mathcal{I}_j(\sigma_{T_i}(j))}$ for each player j in Lines 1–2, which are indexed by all other involved players' sequences $\sigma_{T_i\setminus\{j\}}$ and player j's information set where the last action of his sequence $\sigma_{T_i}(j)$ is taken. Now we can add associated constraints based on these subsets starting from $i = n-2$ (Line 6). For example, in Example 2.1, if we have $BL_{\sigma_1, \textit{J:/ccrc:}} = \{w_1(\sigma_1, \textit{J:/ccrc:c}), w_1(\sigma_1, \textit{J:/ccrc:f})\}$, where $|BL_{\sigma_1, \textit{J:/ccrc:}}| = 2$ is the number of actions in player 2's information set *J:/ccrc:*, then there exists an associated constraint $w_1(\sigma_1, \textit{J:/c:c}) = w_1(\sigma_1, \textit{J:/ccrc:c}) + w_1(\sigma_1, \textit{J:/ccrc:f})$ (we can generate an auxiliary variable $w_1(\sigma_1, \textit{J:/c:c})$ if it does not exist) among these bilinear terms. That is, if player j's sequence $\sigma_{T_i}(j)$ is on the left hand side of bilinear terms in $BL_{\sigma_{T_i \setminus \{j\}}, \mathcal{I}_j(\sigma_j)}$ whose number of terms (corresponding to the number of sequences of player j) is equal to the number of actions in $\mathcal{I}_j(\sigma_j)$ (Line 8), or $BL_{\sigma_{T_i \setminus \{i\}}} (= \{w_{i+1}(\sigma_{T_i \setminus \{i\}}) \mid w_i(\sigma_{T_i}) \in BL_{\sigma_{T_i \setminus \{j\}}, \mathcal{I}_j(\sigma_j)}\}$ representing the set of terms about the right hand side of each bilinear term $w_i(\sigma_{T_i})$) is in the domain of the relation mapping RM (Line 10, initialized by $RM[\{w_{n-1}(\sigma_{n-1}a) \mid a \in \chi(I_{n-1,j})\}] = w_{n-1}(\sigma_{n-1})$ based on Eq.(2.5b) and updated at Line 24), we can add an associated constraint for the equivalence relation (Line 23) between this subset of bilinear terms and the bilinear term involving the sequence reaching this information set (Line 9 or 11). If this term is new, the related subsets and the set of bilinear terms are updated (Lines 15–16). At Lines 17–22, this new bilinear term is connected to one variable if the other variable in this term is certainly 1. The loop (Lines 4–25) will terminate if no new associated constraints are found.

The ARAMDT, remaining to be the upper bound of Problem (2.6), adds associated constraints for the equivalence relation between bilinear terms to the RAMDT (Problem (5.5)), i.e.,

$$\max_{r_1,\ldots,r_{n-1}} v(\mathcal{I}_n(\emptyset))_{ARAMDT} \tag{5.6a}$$

$$\text{Eqs.}(5.5b) - (5.5e) \tag{5.6b}$$

$$\text{Constraints generated by Algorithm 9} \tag{5.6c}$$

Theorem 5.3. *The feasible solution of Problem (2.6) is feasible in Problem (5.6).*

Proof. Suppose the realization plan r_i of player i is the feasible solution of Problem (2.6) but is infeasible in Problem (5.6). By Theorem 5.1, r_i is feasible in Problem (5.5). Therefore, r_i does not satisfy the constraints generated by Algorithm 9. Given $Z_{w_i(\sigma_{T_i})}$ ($\forall w_i(\sigma_{T_i}) \in BL$) in Problem (5.6), we have $|w_i(\sigma_{T_i}) - r_i(\sigma_{T_i}(i)) w_{i+1}(\sigma_{T_{i+1}})| \leq 2^{-Z_{w_i(\sigma_{T_i})}} w_{i+1}(\sigma_{T_{i+1}})$. Now we consider the feasible solutions when $w_i(\sigma_{T_i}) = r_i(\sigma_{T_i}(i)) w_{i+1}(\sigma_{T_{i+1}})$ for all bilinear terms. Without loss of generality, we assume $\sum_{a \in \chi(I_{i,j})} r_i(\sigma_{T_i}(i)a) = r_i(\sigma_{T_i}(i))$, but $\sum_{a \in \chi(I_{i,j})} w_i(\sigma_{T_i}(i)a, \sigma_{T_i \setminus \{i\}}) \neq w_i(\sigma_{T_i})$. We have:

Algorithm 10: Iterative ARAMDT (IARAMDT)

```
1  Initialize $\epsilon_0, \epsilon_1, \epsilon_2, Z_w = Z_r = Z = 0, \forall Z_w, Z_r$;
2  repeat
3      $(r_T, \overline{v}) \leftarrow$ Problem (5.6) with gap $\epsilon_1$;
4      $\underline{v} \leftarrow$ BR, given $r_T$;
5      $Z \leftarrow Z + 1, BL_I \leftarrow \emptyset$;
6      while $BL_I = \emptyset \wedge |\overline{v} - \underline{v}| > \epsilon_0$ do
7          for each $w_i(\sigma_{T_i}) \in BL$ do
8              if $|w_i(\sigma_{T_i}) - r_i(\sigma_i) w_{i+1}(\sigma_{T_{i+1}})| > \epsilon_2$ then
9                  $BL_I \leftarrow BL_I \cup \{w_i(\sigma_{T_i})\}$;
10                 $Z_{w_i(\sigma_{T_i})} \leftarrow Z$ if $Z_{w_i(\sigma_{T_i})} < \lceil -\log_2 \epsilon_2 \rceil$;
11                 $Z_{r_i(\sigma_i)} \leftarrow Z$ if $Z_{r_i(\sigma_i)} < \lceil -\log_2 \epsilon_2 \rceil$;
12         if $BL_I = \emptyset$ then $\epsilon_2 \leftarrow \frac{\epsilon_2}{10}$;
13     Generate a constraint: $v(\mathcal{I}_n(\emptyset))_{ARAMDT} \geq \underline{v}$;
14 until $|\overline{v} - \underline{v}| \leq \epsilon_0$;
15 return $r_T$
```

$$\begin{aligned}
&\textstyle\sum_{a\in\chi(I_{i,j})} w_i(\sigma_{T_i}(i)a, \sigma_{T_i\setminus\{i\}}) \neq w_i(\sigma_{T_i}) \\
\Rightarrow &\sum_{a\in\chi(I_{i,j})} r_i(\sigma_{T_i}(i)a) w_{i+1}(\sigma_{T_i\setminus\{i\}}) \neq r_i(\sigma_{T_i}(i)) w_{i+1}(\sigma_{T_i\setminus\{i\}}) \\
\Rightarrow &\textstyle\sum_{a\in\chi(I_{i,j})} r_i(\sigma_{T_i}(i)a) \neq r_i(\sigma_{T_i}(i))
\end{aligned}$$

which causes a contradiction. Therefore, the feasible solution of Problem (2.6) is feasible in Problem (5.6). □

Theorem 5.4. *$v(\mathcal{I}_n(\emptyset))_{ARAMDT} \geq v(\mathcal{I}_n(\emptyset))$, and $v(\mathcal{I}_n(\emptyset))_{ARAMDT}$ approaches $v(\mathcal{I}_n(\emptyset))$ as $Z_{w_i(\sigma_{T_i})}$ ($\forall w_i(\sigma_{T_i}) \in BL$) approaches ∞ in Problem (5.6).*

Proof. This is immediately obtained from Theorems 5.1–5.3 □

5.3 An Iterative Algorithm for Computing TMEs

Even though the ARAMDT can compute TMEs, it is difficult to decide which precision levels for different bilinear terms could achieve the given accuracy (i.e., ϵ in 'ϵ-TME'). This section develops a novel iterative algorithm to efficiently compute TMEs within any given accuracy based on the ARAMDT. Specifically, we iteratively increase the precision levels for part of bilinear terms until the desired accuracy is achieved, as shown in Algorithm 10. This is based on the fact that the upper bound of the TME value in Problem (5.6) will approach the optimal value when the powers for approximating bilinear terms in the ARAMDT increase by Theorem 5.4. Moreover, given the joint realization plan r_T of the team (i.e., $\times_{i\in T} r_i$) corresponding to the upper bound, we can obtain the lower bound of the optimal value by computing the Best Response (BR) of the adversary. Therefore, we can iteratively increase powers for approximating bilinear terms to make the lower bound and the upper bound tighter to guarantee the accuracy.

Here, the upper bound of the TME value can be computed by solving Problem (5.6), while the lower bound can be obtained through the BR of the adversary, e.g., by using the public-state algorithm [80]. For convenience, we use $Z = 0$ to represent the McCormick relaxation [45], i.e., approximating a bilinear term $w = y_i y_j$ ($y_i, y_j \in [0, 1]$) with loose bounds: $\max\{0, y_i + y_j - 1\} \leq w \leq \min\{y_i, y_j\}$. Line 1 initializes the powers to 0, which means that each bilinear term is approximated by the McCormick relaxation, i.e., constraints of the AMDT in Problem (5.6) are replaced by constraints of the McCormick relaxation. In each iteration, the lower and upper bounds are computed at Lines 3 and 4, respectively. Here, the gap ϵ_1 is set for MILP, i.e., Problem (5.6). After that, the powers are updated at Lines 5–12. Specifically, we increase the powers for bilinear terms (in BL_I) whose difference values between $w_i(\sigma_{T_i})$ and $r_i(\sigma_i)w_{i+1}(\sigma_{T_{i+1}})$ (here $\sigma_i = \sigma_{T_i}(i)$) are larger than ϵ_2 (Lines 8 and 9). The powers for bilinear terms and the variables are updated separately with the bound $\lceil -\log_2 \epsilon_2 \rceil$ (Lines 10 and 11) because $|w_i(\sigma_{T_i}) - r_i(\sigma_i)w_{i+1}(\sigma_{T_{i+1}})| \leq 2^{-Z_{w_i(\sigma_{T_i})}} w_{i+1}(\sigma_{T_{i+1}}) \leq 2^{-Z_{w_i(\sigma_{T_i})}}$. If no bilinear terms fulfill the requirements, ϵ_2 will decrease at Line 12 to guarantee that we can achieve the given accuracy. The lower bound of the objective value of Problem (5.6) is updated at Line 13. The loop will terminate if the gap between the lower bound and the upper bound is not larger than ϵ_0 (Line 14) and then the joint realization plan of team players are returned (Line 15). The following theorem shows that the output r_T is part of an $(\epsilon_0 + \epsilon_1)$-TME.

Theorem 5.5. *The output r_T of Algorithm 10 is part of an $(\epsilon_0 + \epsilon_1)$-TME.*

Proof. When we obtain the output r_T of Algorithm 10, we also obtain the optimal value $\overline{v}$ of Problem (5.6). Given the constraints in Problem (5.6) and constraints in

ϵ	10^{-1}	$\frac{10^{-1}}{2}$	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-1}	$\frac{10^{-1}}{2}$	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}
IARAMDT	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	19s	319s	**649s**	**2460s**
IARMDT-1	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$>$2h			
IARMDT+1	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	41	492s	964s	4786s
IRAMDT	4s	47s	$>$2h					52s	$>$2h					
IARMDT	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	$<$1s	7s	252	714s	4425s
IRMDT	4s	172s	$>$2h					61s	$>$2h					
BARON	30s	690s	$>$10h					30s	$>$10h					
IARAMDT	$<$1s	$<$1s	1s	4s	54s	**133s**	**522s**	$<$1s	$<$1s	$<$1s	35s	141s	**399s**	**3242s**
IARMDT-1	$<$1s	$<$1s	$<$1s	4s	14s	$>$2h		$<$1s	$<$1s	2s	3082s	5982s	$>$2h	
IARMDT+1	$<$1s	$<$1s	1s	4s	50s	707s	707s	$<$1s	$<$1s	1s	19s	100s	1885s	$>$2h
IRAMDT	319s	$>$2h						$>$2h						
IARMDT	$<$1s	$<$1s	2s	5s	65s	$>$2h		$<$1s	$<$1s	1s	39s	320s	919s	$>$2h
IRMDT	436s	$>$2h						$>$2h						
BARON	60s	$>$10h						540s	$>$10h					

TABLE 5.1: The top left part is 3K4 with $|BL| = 120$, the top right part is 3K5 with $|BL| = 200$, the bottom left part is 3K6 with $|BL| = 300$, the bottom right part is 3K7 with $|BL| = 420$, $\Delta_U = 6$, and '$>$ #h': algorithms are terminated after # hours.

Eqs.(2.5a)-(2.5c) for the adversary, $\overline{v}$ is also the optimal objective value in Problem: $\max_{r_T} \min_{r_n} \sum_{\sigma_n \in \Sigma_n} \sum_{\sigma_T \in \Sigma_T} U_T(\sigma_T, \sigma_n) w_1(\sigma_T) r_n(\sigma_n)$. Let r_n be the optimal solution in this problem. Suppose $v^\star$ is the team's utility value under profile (r_T, r_n). We know that $\underline{v} \leq v^\star \leq \overline{v} + \epsilon_1$. Consequently, given r_n, for any strategy r'_T by the unilateral deviation from r_T with the team's utility v', we have $v' \leq \overline{v} + \epsilon_1$ and then $v' - v^\star \leq \overline{v} + \epsilon_1 - \underline{v} \leq \epsilon_0 + \epsilon_1$. Given r_T, for any adversary strategy r'_n with the adversary's utility v', we have $v' \leq -\underline{v}$ and then $v' - (-v^\star) \leq -\underline{v} + \overline{v} + \epsilon_1 \leq \epsilon_0 + \epsilon_1$. Then r_T is part of an $(\epsilon_0 + \epsilon_1)$ NE. In addition, $\overline{v} + \epsilon_1$ is the upper bound of the TME value $v^{\star\star}$ and then $v^{\star\star} - v^\star \leq \overline{v} + \epsilon_1 - v^\star \leq \overline{v} + \epsilon_1 - \underline{v} \leq \epsilon_0 + \epsilon_1$. Therefore, r_T is part of an $(\epsilon_0 + \epsilon_1)$-TME. $\square$

5.4 Experimental Evaluation

We evaluate our algorithm (IARAMDT) through experiments. We use CPLEX (version 12.9) to solve the linear program. All the algorithms are performed on a machine with 6-core 3.2GHz CPU and 16GB memory. Because the default gap for MILPs in CPLEX is 10^{-6}, we set $\epsilon_1 = 10^{-6}$ with $\epsilon_2 = 5 \times 10^{-7}$ unless otherwise specified.

We use BARON (version 19.3.24) [27] to solve Problem (2.6) directly as a baseline. Other baselines include: 1) IARAMDT-1: at each iteration, we only select bilinear terms with the largest difference between $w_i(\sigma_{T_i})$ and $r_i(\sigma_i) w_{i+1}(\sigma_{T_{i+1}})$ similar to the

ϵ	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-1}	10^{-2}	10^{-3}	$\frac{10^{-3}}{2}$
1	2s	**44s**	**481s**	**7047s**	3s	218s	**1831s**	**5591s**
2	2s	>3h			3s	>3h		
3	2s	274s	1303s	>3h	3s	155s	5121s	>3h
4	2s	83s	4442s	>3h	3s	209s	6924s	6924s

TABLE 5.2: The left part is 4K5 with $|BL| = 1840$ and $\Delta_U = 8$, the right part is 3L3 with $|BL| = 1380$ and $\Delta_U = 21$ ($\epsilon_1 = \epsilon_2 = 0.001$), and rows 1–4 are IARAMDT, IARAMDT-1, IARAMDT+1, IARMDT, respectively, which are terminated after 3 hours.

previous approach [44]; 2) IARAMDT+1: we increase the power with 1 instead of Z at lines 10 and 11 of Algorithm 10, which is similar to many approaches [42, 44]; 3) IRAMDT: we use the RAMDT instead of the ARAMDT; 4) IARMDT: we use the MDT to replace the AMDT; and 5) IRMDT: we increase the powers with 1 for all bilinear terms starting with $Z = 1$, which is the original MDT approach [40, 43].

Our experiments run on the standard games, the multiplayer Kuhn poker and the Leduc Hold'em poker games (see Farina et al. (2018) for their rules), where nKr and nLr denote an n-player Kuhn instance with r ranks (i.e., r cards) and an n-player Leduc instance with r ranks (i.e., $2r$ cards), respectively. Δ_U is the difference between the maximum possible utility and the minimum possible utility of the team. In addition to the current game rules, the game ends if the adversary takes the action of folding because the team will certainly win after that. Without loss of generality, player n is the adversary.

We show the time to compute $\epsilon\Delta_U$-TMEs for different accuracies. Here $\epsilon\Delta_U$ is $\epsilon_0 + \epsilon_1$ in Algorithm 10 or the difference between the lower bound and the upper bound in BARON. Results in Table 5.1 show that 1) IARAMDT dramatically outperforms the exiting MDT approach and BARON; 2) our approximation techniques, especially the associated constraint method, are extremely important for the scalability of IARAMDT by comparing it with IRAMDT and IARMDT; and 3) our novel techniques in our iterative algorithm improve the scalability of IARAMDT by comparing it with IARAMDT-1 and IARAMDT+1. Specifically, after using our associated constraint method with the same initial setting (i.e., using the McCormick relaxation) in the iterative algorithm, IARAMDT-1, IARAMDT+1, and IARMDT perform similarly to IARAMDT when we

ϵ	10^{-1}	10^{-2}	10^{-3}	10^{-6}	10^{-1}	10^{-2}	10^{-3}	10^{-6}
1	**74%**	**0%**	**0%**	**0%**	**60%**	**60%**	**10%**	**0%**
5	<1s	3s	6s	6s	5s	38s	169s	169s
5	*75%*	*70%*	*80%*	*80%*	*76%*	*80%*	*84%*	*84%*
6	<1s	4s	>2h		<1s	10s	>2h	
6	*87%*	*69%*			*91%*	*69%*		
1	*86%*	**20%**	**17%**	**0%**	**77%**	**67%**	**12%**	**0%**
5	24s	201s	1305s	2208s	59s	1154s	>2h	
5	*84%*	*86%*	*84%*	*82%*	*83%*	*82%*		
6	<1s	7s	>2h		<1s	23s	>2h	
6	*88%*	*76%*			*93%*	*77%*		

TABLE 5.3: The gaps and runtime of IARAMDT (label 1) (see Table 5.1 for its runtime), HCG (label 5), and FTP (label 6) on 3K4, 3K5, 3K6, and 3K7, respectively.

only need low precision levels to achieve the given accuracy, but they perform significantly worse than IARAMDT when we need high precision levels to achieve the given accuracy, which is further confirmed in larger problems as shown in Table 5.2.

Comparison to TMSP. Table 5.3 shows the comparison to the TMSP obtained from an $\epsilon\Delta_U$-TMECor, which is computed by algorithm Hybrid Column Generation (HCG) [15] or algorithm Fictitious Team-Play (FTP) [28]. The gap is the relative distance between the team's best utility (v_m) of IARAMDT (the lower bound when $\epsilon = 10^{-6}$) and the team's lower bound (v) under different values of ϵ, i.e., $\frac{|v-v_m|}{|v|} \times 100\%$. The larger the gap is, the more utility the team will lose. We can see that: 1) the gaps of IARAMDT monotonically decrease with ϵ to 0, but the gaps of the TMSP may not decrease with ϵ and are always very large; and 2) given the same ϵ, IARAMDT runs faster than HCG and FTP in most cases. About the TMSP obtained from a TMECom, it can be computed within 1s in four games of Table 5.3, but the gaps (i.e., 88%, 94%, 95%, and 92%) are extremely large. Therefore, the TMSP can cause a huge loss, which is consistent with our theoretical result shown in Proposition 5.1.

5.5 Chapter Summary

This chapter proposes an efficient algorithm to compute TMEs within any given accuracy for zero-sum multiplayer EFGs. We first show that the inefficiency of correlated strategies can be arbitrarily large in EFGs. To efficiently solve the non-convex program for finding TMEs directly, we develop novel approximation techniques (i.e., an asynchronous precision method and an associated constraint method) and the novel iterative algorithm. Our algorithm is dramatically faster than baselines in the experimental evaluation.

Chapter 6

Computing *Ex Ante* Coordinated Team-Maxmin Equilibria in Zero-Sum Multiplayer Extensive-Form Games

This chapter[1] focuses on computing TMEsCor [15, 28] in zero-sum multiplayer extensive-form games. To compute TMEsCor efficiently, we first propose a novel hybrid-form strategy representation for the team, where one team member takes sequence-form strategies while other team members take normal-form strategies. Second, based on our hybrid-form strategies, we develop our column generation algorithm with two novel features: 1) we provide a finite upper bound for the number of iterations where our algorithm terminates even though the strategy space is infinite; and 2) we design a novel BR for the team, which involves multilinear terms representing reaching probabilities for terminal nodes to reduce the number of variables. Third, we develop our novel associated representation technique for solving our multilinear BR, which has two novel features: 1) exactly representing multilinear terms by linear constraints and 2) efficiently generating associated constraints for the equivalence relation between multilinear terms in the game tree. Finally, extensive experiments show that our algorithm is several orders of magnitude faster than state-of-the-art algorithms in large games. Thus,

[1]The work in this chapter has been submitted as **Youzhi Zhang** and Bo An. Computing *ex ante* coordinated team-maxmin equilibria in zero-sum multiplayer extensive-form games. *Under review by the 34th Conference on Neural Information Processing Systems (**NeurIPS**)*, 2020.

this chapter shows that, to compute equilibria, the approach formulating the problem as a multilinear program with global optimization techniques can be dramatically faster than the approach immediately formulating it as a linear program.

6.1 A New Scalable Approach

Even though TMEsCor can be computed via a Linear Program (LP) by enumerating all strategies in Π_T [15], it is still inefficient to compute TMEsCor because Π_i is exponential in the size of the game tree. Column Generation (CG) is a standard approach to conquer this difficulty [15]. However, the Best Response (BR) oracle for the team in CG is usually a Mixed-Integer LP (MILP) involving a large number of integer variables, which makes CG hard to converge to a TMECor in large games. The fundamental reason for the large number of integer variables in BR is that BR needs to generate a joint pure NFS as a best response and needs to represent a joint pure NFS's reaching probability for each terminal node (e.g., using integer 1 or 0 to represent reaching or not, and thus having $|L|$ integer variables [15]). To reduce the number of variables, our key idea is that: 1) we do not generate pure NFSs for all team members; and 2) we use multilinear terms to represent the team's reaching probabilities for terminal nodes in BR. To do that, we provide two novel methods: 1) a novel Hybrid-Form Strategy (HFS) representation for the team, where one team member takes an SFS (it is in a continuous space) while other team members take pure NFSs[1]; and 2) our novel BR with only $\sum_{i\in T\setminus\{1\}}|\Sigma_i|$ integer variables based on HFSs[2] (using pure SFSs to represent pure NFSs in an HFS), which involves multilinear terms to represent an HFS's reaching probabilities for terminal nodes. Moreover, we provide two novel solutions to make our approach applicable: 1) we provide a finite upper bound for the number of iterations where CG terminates, even though the strategy space is infinite; and 2) we develop our novel Associated Representation Technique (ART) to exactly and efficiently solve our multilinear BR.

[1]If all team players take pure NFSs, our algorithm can work as well but can be significantly slower than the algorithm with HFSs as shown in our experiments later. In addition, if there are two or more team players taking SFSs, reaching probabilities for terminal nodes cannot be exactly represented by linear constraints anymore.

[2]We can choose any team player to take the SFS, and we choose player 1 just for convenience here.

6.1.1 Hybrid-Form Strategies for the Team

Now we propose a new hybrid-form strategy representation for the team, where one team member takes the sequence-form strategy while other team members take pure normal-form strategies. A pure hybrid-form strategy for the team is defined by a tuple:

$$f_T = (r_1, \pi_{T\setminus\{1\}})$$

where $r_1 \in \mathcal{R}_1$ and $\pi_{T\setminus\{1\}} \in \Pi_{T\setminus\{1\}} = \times_{i\in T\setminus\{1\}}\Pi_i$. Given a strategy profile (f_T, σ_n), similar to the definition of $U_T(\pi_T, \sigma_n)$ in Eq.(2.8) with $L_{f_T,\sigma_n}(\subseteq L)$ reached by (f_T, σ_n), the team's utility is:

$$U_T(f_T, \sigma_n) = \textstyle\sum_{l\in L_{f_T,\sigma_n}} r_1(\text{seq}_1(l))u_T(l)c(l)$$

Let $\mathcal{F}_T$ be the set of pure hybrid-form strategies and $\overline{x}_T$ be a mixed hybrid-form strategy strategy, i.e., a probability distribution over $\mathcal{F}_T$ ($\overline{x}_T \in \Delta(\mathcal{F}_T)$). The team's expected utility is:

$$U_T(\overline{x}_T, \sigma_n) = \textstyle\sum_{f_T\in\mathcal{F}_T} U_T(f_T, \sigma_n)\overline{x}_T(f_T)$$

Similarly, $U_T(\overline{x}_T, r_n) = \sum_{f_T\in\mathcal{F}_T} U_T(f_T, r_n)\overline{x}_T(f_T)$ and

$$U_T(f_T, r_n) = \sum_{l\in L_{f_T,r_n}} r_n(\text{seq}_n(l))r_1(\text{seq}_1(l))u_T(l)c(l)$$

We define the support sets in a mixed normal-form strategy and a mixed hybrid-form strategy:

$$S_{x_T} = \{\pi_T \mid x_T(\pi_T) > 0, \pi_T \in \Pi_T\}, S_{\overline{x}_T} = \{f_T \mid \overline{x}_T(f_T) > 0, f_T \in \mathcal{F}_T\}$$

Two team's strategies are realization-equivalent if both define the same probabilities for reaching nodes given any strategies of the adversary. Now we show the realization equivalence between hybrid-form strategies and normal-form strategies [1].

[1] $x_T \in \Delta(\Pi_T)$ in this chapter.

Lemma 6.1. *$\forall x_T \in \Delta(\Pi_T)$, $\exists \overline{x}_T \in \Delta(\mathcal{F}_T)$ such that $x_T \sim \overline{x}_T$ with $|S_{x_T}| = |S_{\overline{x}_T}|$.*

Proof. As we mentioned in Section 2.1.2, any pure normal-form strategy $\pi_1 \in \Pi_1$ can be uniquely represented by a pure sequence-form strategy $r_1 \in \overline{\mathcal{R}}_1$, i.e., any pure normal-form strategy $\pi_1 \in \Pi_1$ is realization-equivalent to a pure sequence-form strategy $r_1 \in \overline{\mathcal{R}}_1$. Obviously, if $\pi_1 \sim r_1$,

$$(\pi_1, \pi_{T\setminus\{1\}}) \sim (r_1, \pi_{T\setminus\{1\}})$$

where $\pi_1 \in \Pi_1$ and $r_1 \in \overline{\mathcal{R}}_1$. For any mixed normal-form strategy $x_T \in \Delta(\Pi_T)$, we define a mixed hybrid-form strategy $\overline{x}_T \in \Delta(\mathcal{F}_T)$ such that:

$$\overline{x}_T(r_1, \pi_{T\setminus\{1\}}) = x_T(\pi_1, \pi_{T\setminus\{1\}}) \quad \forall(\pi_1, \pi_{T\setminus\{1\}}) \in \Pi_T, \pi_1 \sim r_1, r_1 \in \overline{\mathcal{R}}_1$$

and $\overline{x}_T(f_T) = 0$ for other strategies $f_T \in \mathcal{F}_T \setminus \{(r_1, \pi_{T\setminus\{1\}}) \mid r_1 \in \overline{\mathcal{R}}_1\}$. Given any adversary strategy r_n (or σ_n) and any node $h \in H \cup L$, let $P_{\pi_T}(h)$ be π_T's reaching probability for h. Then x_T's reaching probability for h is:

$$\begin{aligned} &\sum_{(\pi_1,\pi_{T\setminus\{1\}})\in\Pi_T} x_T(\pi_1, \pi_{T\setminus\{1\}}) P_{(\pi_1,\pi_{T\setminus\{1\}})}(h) \\ = &\sum_{(r_1,\pi_{T\setminus\{1\}})\in\mathcal{F}_T, r_1\sim\pi_1} \overline{x}_T(r_1, \pi_{T\setminus\{1\}}) P_{(\pi_1,\pi_{T\setminus\{1\}})}(h) \end{aligned}$$

which is $\overline{x}_T$'s reaching probability for h. Hence, $x_T \sim \overline{x}_T$ and $|S_{x_T}| = |S_{\overline{x}_T}|$. □

Lemma 6.2. *$\forall \overline{x}_T \in \Delta(\mathcal{F}_T)$,$\exists x_T \in \Delta(\Pi_T)$ such that $x_T \sim \overline{x}_T$.*

Proof. As we mentioned in Section 2.1.2, any sequence-form strategy $r_1 \in \mathcal{R}_1$ is realization-equivalent to mixed a normal-form strategy $x_1 \in \Delta(\Pi_1)$. Obviously, if $x_1 \sim r_1$,

$$(x_1, \pi_{T\setminus\{1\}}) \sim (r_1, \pi_{T\setminus\{1\}})$$

where $x_1 \in \Delta(\Pi_1)$ and $r_1 \in \mathcal{R}_1$. For any mixed hybrid-form strategy $\overline{x}_T \in \Delta(\mathcal{F}_T)$, we define a mixed normal-form strategy $x_T \in \Delta(\Pi_T)$ such that:

$$x_T(\pi_1, \pi_{T\setminus\{1\}}) = \sum_{(r_1, \pi_{T\setminus\{1\}}) \in \mathcal{F}_T, x_1 \sim r_1} \overline{x}_T(r_1, \pi_{T\setminus\{1\}}) x_1(\pi_1) \quad \forall (\pi_1, \pi_{T\setminus\{1\}}) \in \Pi_T$$

Given any adversary strategy r_n (or σ_n) and any node $h \in H \cup L$, let $P_{f_T}(h)$ be f_T's reaching probability for h. We have: if $x_1 \sim r_1$,

$$P_{(r_1, \pi_{T\setminus\{1\}})}(h) = \sum_{\pi_1 \in \Pi_1} x_1(\pi_1) P_{(\pi_1, \pi_{T\setminus\{1\}})}(h)$$

Then $\overline{x}_T$'s reaching probability for h is:

$$\begin{aligned}
&\sum_{(r_1, \pi_{T\setminus\{1\}}) \in \mathcal{F}_T} \overline{x}_T(r_1, \pi_{T\setminus\{1\}}) P_{(r_1, \pi_{T\setminus\{1\}})}(h) \\
= &\sum_{(r_1, \pi_{T\setminus\{1\}}) \in \mathcal{F}_T, r_1 \sim x_1} \overline{x}_T(r_1, \pi_{T\setminus\{1\}}) \sum_{\pi_1 \in \Pi_1} x_1(\pi_1) P_{(\pi_1, \pi_{T\setminus\{1\}})}(h) \\
= &\sum_{(\pi_1, \pi_{T\setminus\{1\}}) \in \Pi_T} \sum_{(r_1, \pi_{T\setminus\{1\}}) \in \mathcal{F}_T, r_1 \sim x_1} \overline{x}_T(r_1, \pi_{T\setminus\{1\}}) x_1(\pi_1) P_{(\pi_1, \pi_{T\setminus\{1\}})}(h) \\
= &\sum_{(\pi_1, \pi_{T\setminus\{1\}}) \in \Pi_T} x_T(\pi_1, \pi_{T\setminus\{1\}}) P_{(\pi_1, \pi_{T\setminus\{1\}})}(h)
\end{aligned}$$

which is x_T's reaching probability for h. Hence, $x_T \sim \overline{x}_T$. □

Theorem 6.1. *$\forall x_T \in \Delta(\Pi_T)$, $\exists \overline{x}_T \in \Delta(\mathcal{F}_T)$ such that $U_T(x_T, \sigma_n) = U_T(\overline{x}_T, \sigma_n)(\forall \sigma_n \in \Sigma_n)$, and vice versa.*

Proof. By Lemma 6.1, $\forall x_T \in \Delta(\Pi_T)$, $\exists \overline{x}_T \in \Delta(\mathcal{F}_T)$ such that $x_T \sim \overline{x}_T$. For each $\sigma_n \in \Sigma_n$, let $p_{\sigma_n}(l)$ be the probability reaching terminal node l by x_T or $\overline{x}_T$. Then,

$$\begin{aligned}
&U_T(\overline{x}_T, \sigma_n) \\
= &\sum_{f_T \in \mathcal{F}_T} U_T(f_T, \sigma_n) \overline{x}_T(f_T) \\
= &\sum_{l \in L} p_{\sigma_n}(l) u_T(l) c(l) \\
= &\sum_{\pi_T \in \Pi_T} U_T(\pi_T, \sigma_n) x_T(\pi_T)
\end{aligned}$$

Algorithm 11: CMB

```
1 Initialize $\mathcal{F}'_T \leftarrow \{\text{any strategy}\}, \underline{v} \leftarrow 0, \overline{v} \leftarrow 1$;
2 repeat
3     $(\underline{v}, \overline{x}_T, r_n) \leftarrow$ solution of Problem (6.1) under $(\mathcal{F}'_T, \Sigma_n)$;
4     $(\overline{v}, f_T) \leftarrow BR(r_n)$;
5     $\mathcal{F}'_T \leftarrow \mathcal{F}'_T \cup \{f_T\}$;
6 until $\underline{v} = \overline{v}$;
7 return $(\overline{x}_T, r_n)$.
```

$$=U_T(x_T, \sigma_n)$$

Similarly, by Lemma 6.2, $\forall \overline{x}_T \in \Delta(\mathcal{F}_T)$, $\exists x_T \in \Delta(\Pi_T)$ such that $U_T(x_T, \sigma_n) = U_T(\overline{x}_T, \sigma_n)(\forall \sigma_n \in \Sigma_n)$. □

Therefore, the set of TMEsCor will not change if HFSs are played, and a TMECor can be computed by:

$$\max_x v(\mathcal{I}_n(\emptyset)) \tag{6.1a}$$

$$v(\mathcal{I}_n(\sigma_n)) - \sum_{I_{n,j} \in I_n : \text{seq}_n(I_{n,j}) = \sigma_n} v(I_{n,j}) \leq U_T(\overline{x}_T, \sigma_n) \quad \forall \sigma_n \in \Sigma_n \tag{6.1b}$$

$$\sum_{f_T \in \mathcal{F}_T} \overline{x}_T(f_T) = 1 \tag{6.1c}$$

$$\overline{x}_T(f_T) \geq 0 \quad \forall \pi_T \in \mathcal{F}_T \tag{6.1d}$$

where $\mathcal{I}_n(\sigma_n)$ defines an information set in which player n takes the last action of sequence σ_n, and $v(I_{n,j})$ defines the expected utility of the team in each information set $I_{n,j}$. The adversary takes the strategy minimizing the team's utility in each information set $\mathcal{I}_n(\sigma_n)$, represented by Eq.(6.1b).

6.1.2 CG with a Multilinear BR

It is impractical to enumerate all HFSs for solving Problem (6.1) because $\mathcal{F}_T$ is continuous due to one player's SFS in an HFS. In this section, we develop our CG with a Multilinear BR (CMB) with two novel features: 1) we provide a finite upper bound for the number of iterations where CMB terminates, even though the strategy space is infinite; and 2) we design our novel multilinear BR for the team.

CMB is shown in Algorithm 11. Starting from the restricted $\mathcal{F}'_T$ (initialized at Line 1), Program (6.1) solves the equilibrium $(\overline{x}_T, r_n)$ with utility $\underline{v}$ for the team (Line 3). CMB then finds the team's best response f_T with utility $\overline{v}$ for the team by calling BR (Line 4) and expands $\mathcal{F}'_T$ (Line 5). CMB can find an equilibrium due to properties shown in the following theorems: 1) CMB will terminate within finite iterations; and 2) its output is a TMECor when it terminates.

Unfortunately, the guarantee of convergence of CMB usually needs to assume that the strategy space is finite to make sure that CMB can terminate when it needs to enumerate pure strategies in the worst case [5, 38]. If CMB needs to enumerate all pure strategies in an infinite space, then CMB will not converge anymore. To make sure that we can use our hybrid-form representation to compute exact TMEsCor, we theoretically show that CMB can guarantee to converge within a finite number of iterations in our infinite strategy space. That is, we show that this number of iterations can be bounded as follows:

Theorem 6.2. *The number of iterations where CMB terminates is at most* $2^{|\Pi_1|}\prod_{i\in T\setminus\{1\}}|\Pi_i|$.

Proof. Given the adversary strategy r_n, suppose that $f_T = (r_1, \pi_{T\setminus\{1\}})$ is a best response for the team. Let x_1 be a realization-equivalent normal-form strategy of r_1 with its support set $S_{x_1} = \{\pi_1 \mid x_1(\pi_1) > 0, \pi_1 \in \Pi_1\}$. Then, $(x_1, \pi_{T\setminus\{1\}})$ and f_T are realization-equivalent with:

$$\begin{aligned}&U_T(f_T, r_n)\\=&U_T((x_1, \pi_{T\setminus\{1\}}), r_n)\\=&\sum_{\pi_1\in S_{x_1}} x_1(\pi_1)U_T(\pi_1, \pi_{T\setminus\{1\}}, r_n)\end{aligned}$$

Note that, if $(x_1, \pi_{T\setminus\{1\}})$ is a best response against r_n, then $(\pi_1, \pi_{T\setminus\{1\}})(\forall \pi_1 \in S_{x_1})$ is also a best response against r_n. Then, for any $\pi_1 \in S_{x_1}$ we have $U_T(\pi_1, \pi_{T\setminus\{1\}}, r_n) = U_T(f_T, r_n)$.[1] Therefore, for any $r'_1 \in \mathcal{R}_1$ with a realization-equivalent normal-form

[1] If there is $\pi_1 \in S_{x_1}$ such that $U_T(\pi_1, \pi_{T\setminus\{1\}}, r_n) < U_T((x_1, \pi_{T\setminus\{1\}}), r_n)$, then there is $\pi'_1 \in S_{x_1}$ such that $U_T(\pi'_1, \pi_{T\setminus\{1\}}, r_n) > U_T((x_1, \pi_{T\setminus\{1\}}), r_n)$, and then higher $U_T((x_1, \pi_{T\setminus\{1\}}), r_n)$ is achieved by moving the probability for π_1 to π'_1, which causes a contradiction.

strategy x'_1 such that $S_{x'_1} = S_{x_1}$,

$$\begin{aligned} & U_T(r'_1, \pi_{T\setminus\{1\}}, r_n) \\ =& \sum_{\pi_1 \in S_{x_1}} x'_1(\pi_1) U_T(\pi_1, \pi_{T\setminus\{1\}}, r_n) \\ =& U_T(x_1, \pi_{T\setminus\{1\}}, r_n) \\ =& U_T(f_T, r_n) \end{aligned}$$

For each $\Pi'_1 \subseteq \Pi_1$, we can define a set of pure hybrid-form strategies:

$$\mathcal{F}_T(\Pi'_1, \pi_{T\setminus\{1\}}) = \{(r_1, \pi_{T\setminus\{1\}}) \in \mathcal{F}_T \mid S_{x_1} = \Pi'_1(\subseteq \Pi_1), x_1 \sim r_1\}$$

where $\forall f_T \in \mathcal{F}_T(\Pi'_1, \pi_{T\setminus\{1\}})$, f_T is a best response against r_n if $\exists\, f'_T \in \mathcal{F}_T(\Pi'_1, \pi_{T\setminus\{1\}})$ is a best response against r_n. In CMB, only the best response against r_n, which must be better than any strategy in $\mathcal{F}'_T$ against r_n, will be added to $\mathcal{F}'_T$. Therefore, only one strategy f'_T in $\mathcal{F}_T(\Pi'_1, \pi_{T\setminus\{1\}})$ will be added to $\mathcal{F}'_T$, i.e., the set of pure hybrid-form strategies $\mathcal{F}_T(\Pi'_1, \pi_{T\setminus\{1\}})$ is represented by a single strategy f'_T in $\mathcal{F}'_T$. In the worst case, CMB needs to add strategies involving all these sets (the number of these sets is up to $2^{|\Pi_1|} \prod_{i \in T\setminus\{1\}} |\Pi_i|$), and then CMB will terminate within $2^{|\Pi_1|} \prod_{i \in T\setminus\{1\}} |\Pi_i|$ iterations. □

The above result implies that there always exists a TMECor with a finite-sized support set for the team, which is rather loose. Now we provide a stronger bound on the size of the team's support set.

Theorem 6.3. *There is TMECor $(\overline{x}_T, r_n)$ with $|S_{\overline{x}_T}| \leq |\Sigma_n|$.*

Proof. There is TMECor (x_T, r_n) with $|S_{x_T}| \leq |\Sigma_n|$ [15]. We know that there is $\overline{x} \in \Delta(\mathcal{F}_T)$ with $|S_{x_T}| = |S_{\overline{x}_T}|$ such that $x_T \sim \overline{x}_T$ by Lemma 6.1. By Theorem 6.1, $U_T(x_T, \sigma_n) = U_T(\overline{x}_T, \sigma_n)(\forall \sigma_n \in \Sigma_n)$. Then, $(\overline{x}_T, r_n)$ is a TMECor, concluding the proof. □

Our experiments show that the actual number of iterations where CMB terminates is significantly smaller than the bound in Theorem 6.2, which could be due to the fact that at least one TMECor has a small support set shown in Theorem 6.3. Now we show that the output of CMB is a TMECor.

Theorem 6.4. *CMB converges to a TMECor.*

Proof. By Theorem 6.2, CMB will terminate with $\overline{v} = \underline{v}$ within a finite number of iterations. Then, we have, $U_T(\overline{x}_T, r_n) = \underline{v} = \overline{v} \geq U_T(\overline{x}'_T, r_n)(\forall \overline{x}'_T)$, and $-U_T(\overline{x}_T, r_n) = -\underline{v} \geq -U_T(\overline{x}_T, r'_n)(\forall r'_n)$. Therefore, $(\overline{x}_T, r_n)$ is a TMECor. $\square$

In addition to exact TMEsCor, CMB converges to an approximate TMECor if we change the termination condition from $\overline{v} = \underline{v}$ to $\overline{v} - \underline{v} \leq \epsilon$.

Theorem 6.5. *If CMB terminates with $\overline{v} - \underline{v} \leq \epsilon$, then its output $(\overline{x}_T, r_n)$ is an ϵ-TMECor.*

Proof. $U_T(\overline{x}'_T, r_n) - U_T(\overline{x}_T, r_n) \leq \overline{v} - \underline{v} \leq \epsilon$ $(\forall \overline{x}'_T)$, and $-U_T(\overline{x}_T, r'_n) - (-U_T(\overline{x}_T, r_n)) \leq -\underline{v} - (-\underline{v}) = 0(\forall r'_n)$. Therefore, $(\overline{x}_T, r_n)$ is an ϵ-TMECor. $\square$

Now we propose our multilinear BR to compute the best response by the following multilinear program:

$$\max_{\times_{i\in T} r_i} \textstyle\sum_{l\in L} u_T(l)c(l)r_n(\text{seq}_n(l))\prod_{i\in T} r_i(\text{seq}_i(l)) \tag{6.2a}$$

$$\text{Eqs.}(2.5a) - (2.5c) \quad \forall i \in T \tag{6.2b}$$

$$r_i(\sigma_i) \in \{0, 1\} \quad \forall \sigma_i \in \Sigma_i, i \in T \setminus \{1\} \tag{6.2c}$$

$$r_1(\sigma_1) \in [0, 1] \quad \forall \sigma_1 \in \Sigma_1 \tag{6.2d}$$

where $r_i \in \overline{\mathcal{R}}_i$ represents its realization-equivalent $\pi_i \in \Pi_i$ $(i \in T \setminus \{1\})$ in an HFS. Basically, our BR considers the probability of all players reaching each terminal node, and then involves multilinear term $\prod_{i\in T} r_i(\text{seq}_i(l))$. However, it only has $\sum_{i\in T\setminus\{1\}} |\Sigma_i|$ integer variables.

6.1.3 Associated Representation Technique (ART)

This section proposes our novel technique to solve the multilinear Problem (6.2) efficiently by developing the ART to exactly represent multilinear terms in Problem (6.2) through linear constraints. The ART is inspired by the recent techniques to approximate multilinear terms [16], which recursively transforms multilinear terms into bilinear terms and then uses an MILP to approximate each bilinear term while exploiting the equivalence relation between bilinear terms. Moreover, the ART has two novel features: 1) exactly representing multilinear terms without using the recursive method, and 2) efficiently generating associated constraints for the equivalence relation between multilinear terms in the game tree.

6.1.3.1 Multilinear Representation

To transform Problem (6.2) to an MILP, based on the property of integer variables in Problem (6.2), we develop our technique, Multilinear Representation (MR), exactly representing multilinear terms without introducing new integer variables. That is, for each multilinear term $w(\sigma_T) = \prod_{i\in T} r_i(\sigma_T(i))$, where $\sigma_T(i)$ is the sequence of player i in joint sequence $\sigma_T \in \Sigma_T(i.e., \times_{i\in T}\Sigma_i)$, $r_i \in \overline{\mathcal{R}}(i \in T \setminus \{1\})$, and $r_1 \in \mathcal{R}_1$, we have:

$$0 \le w(\sigma_T) \le r_i(\sigma_T(i)) \quad \forall i \in T \setminus \{1\} \tag{6.3a}$$

$$0 \le r_1(\sigma_T(1)) - w(\sigma_T) \le n - 2 - \textstyle\sum_{i\in T\setminus\{1\}} r_i(\sigma_T(i)) \tag{6.3b}$$

Theorem 6.6. *$w(\sigma_T)$ is exactly represented by Eqs.(6.3a) and (6.3b).*

Proof. $w(\sigma_T) = r_1(\sigma_T(1))$ if $r_i(\sigma_T(i)) = 1(\forall i \in T \setminus \{1\})$, otherwise $w(\sigma_T) = 0$. First, by Eq.(6.3b), $0 \le r_1(\sigma_T(1)) - w(\sigma_T) \le n - 2 - (n-2) = 0$ if $r_i(\sigma_T(i)) = 1(\forall i \in T \setminus \{1\})$, i.e., $w(\sigma_T) = r_1(\sigma_T(1))$. Second, by Eq.(6.3a), $w(\sigma_T) = 0$ if any $i \in T \setminus \{1\}$ with $r_i(\sigma_T(i)) = 0$. Then, $w(\sigma_T)$ is exactly represented by Eqs.(6.3a) and (6.3b). $\square$

By using MR, we can exactly transform Problem (6.2) to an MILP without introducing new integer variables.

6.1.3.2 Efficient Associated Constraint Generation

An MILP is generally solved by the branch-and-bound approach with the LP relaxation [81]. In MR, the relaxation may give us a much larger feasible solution space because $w(\sigma_T)$ may not be exactly represented by Eqs.(6.3a) and (6.3b) after the relaxation of $r_i(\sigma_T(i))(i \in T \setminus \{1\})$. Now we aim to reduce this feasible solution space and solve the MILP efficiently. To do that, we add associated constraints by exploiting the equivalence relation between multilinear terms.

The equivalence relation between multilinear terms is based on the constraints for the SFS in Eqs.(2.5a)-(2.5c), as shown in the following example.

Example 6.1. *In a four-player Kuhn poker game, the information set for player i includes the information about the card that player i holds and the observed actions taken by players in turn. For example, J:/cccr: is an information set of player 1, where player 1 has card J and has observed the actions 'c,c,c,r' of all players in turn. Now, player 1 has two actions: calling (c) and folding (f) in this information set J:/cccr: reached by sequence J:/:c of player* 1*. Therefore, by Eq.(2.5b), we have $r_1($J:/:c$) = r_1($J:/cccr:c$) + r_1($J:/cccr:f$)$. Moreover, this information set may include many different nodes because of the uncertainty about the holding cards of other players. Each node in the information set defines a sequence for each player from the root to this node. For example $($J:/:c, Q:/c:c, T:/cc:c, K:/ccc:r$)$ and $($J:/:c, Q:/c:c, K:/cc:c, T:/ccc:r$)$ are two different nodes in this information set due to different cards for players 3 and 4. Therefore, there are many different joint sequences for the team involved by an information set, e.g., $($J:/:c, Q:/c:c, T:/cc:c$)$ or $($J:/:c, Q:/c:c, K:/cc:c$)$. Player 1 has two actions in this information set and only Player 1 will take actions in this information set. Then, given a joint sequence $($J:/:c, Q:/c:c, T:/cc:c$)$ in this information set, we certainly have succeeding joint sequences $($J:/cccr:c, Q:/c:c, T:/cc:c$)$ and $($J:/cccr:f, Q:/c:c, T:/cc:c$)$ reaching succeeding nodes of this information set. Then, we can have $w($J:/:c, Q:/c:c, T:/cc:c$) = w($J:/cccr:c, Q:/c:c, T:/cc:c$) + w($J:/cccr:f, Q:/c:c, T:/cc:c$)$ because of $r_1($J:/c:c$) = r_1($J:/cccr:c$) + r_1($J:/cccr:f$)$. Moreover, given $w($J:/:c, Q:/c:c, T:/cc:r$)$, we have $w($J:/:c, Q:/c:c, T:/cc:r$) + w($J:/:c, Q:/c:c, T:/cc:c$) = w($J:/:c, Q:/c:c$, \emptyset)$. In addition, given $w($J:/:c, Q:/c:r$, \emptyset)$, we have $w($J:/:c, Q:/c:c$, \emptyset) + w($J:/:c, Q:/c:r$, \emptyset) = w($J:/:c$, \emptyset, \emptyset) = r_1($J:/:c$)$.*

That is, we look for the equivalence relation between multilinear terms until these terms are connected to a sequence's probability in the SFS, which we call associated constraints. Due to these associated constraints, we can immediately rule out the solutions that cannot satisfy these constraints and then solve the MILP efficiently.

Now we develop a simple and efficient algorithm to generate associated constraints, which only enumerates joint sequences in $\Sigma_T(I_{i,j})$ (the set of joint sequences in $I_{i,j}$) by:

$$w(\sigma_T) = \sum_{a\in\chi(I_{i,j})} w(\ldots, \sigma_T(i-1), \sigma_T(i)a, \sigma_T(i+1), \ldots) \\ \forall \sigma_T \in \Sigma_T(I_{i,j}), I_{i,j} \in I_i, i \in T \tag{6.4}$$

Therefore, this new algorithm only runs in time $O(\sum_{i\in T}\sum_{I_{i,j}\in I_i} |\Sigma_T(I_{i,j})|)$. Our algorithm is significantly faster than the existing algorithm [16] because their algorithm conducts too many redundant operations: 1) it generates associated constraints from the terminate nodes to the root, which needs to generate new terms for sequences reaching nonterminal nodes frequently; and 2) it recursively generates associated constraints: it transforms each multilinear term to a set of bilinear terms (e.g., the multilinear term $w($*J:/:c, Q:/c:c, T:/cc:c*$)$ is transformed to two bilinear terms $w($*J:/:c*$, w($Q:/c:c, T:/cc:c$))$ and $w($Q:/c:c, T:/cc:c$))$ and frequently checks whether the equivalence relation between bilinear terms exists.

After adding constraints in Eqs.(6.3a)-(6.3b) and (6.4), we can exactly transform Problem (6.2) to the following MILP:

$$\max_{\times_{i\in T} r_i} \sum_{l\in L} u_T(l)c(l)r_n(\text{seq}_n(l))w(\times_{i\in T}\text{seq}_i(l)) \tag{6.5a}$$

$$\text{Eqs.}(6.2b) - (6.2d), (6.4) \tag{6.5b}$$

$$\text{Eqs.}(6.3a) - (6.3b) \quad \forall w(\times_{i\in T}\text{seq}_i(l)), l \in L \tag{6.5c}$$

Theorem 6.7. *The feasible solution of Problem (6.2) is feasible in Problem (6.5).*

Proof. Suppose that an SFS r_i of player i is feasible in Problem (6.2) but is infeasible in Problem (6.5). By Theorem 6.6, $w(\times_{i\in T}\text{seq}_i(l))$ is exactly represented by Eqs.(6.3a) and (6.3b). Therefore, r_i does not satisfy the constraints in Eq.(6.4).

EFG	$\|L\|$	$\|\Sigma_i\|$	CMB	CMB/H	CMB/A	CMB/ART	CMB/ART/H	C18	F18
3K4	312	33	**0.7s**	0.7s	2.1s	4s	6s	6.8s	1.2s
3K6	1560	49	**2s**	2s	20s	191s	479s	>5h	12s
3K8	4368	65	**4s**	4s	497s	7160s	>5h		210s
3K10	9360	81	**5s**	6s	10530s	>5h			3541s
3K12	17160	97	**10s**	10s	>5h				>5h
$4L3_1$	30600	219	**68s**	165s					
$4L3_2$	638064	219	**1264s**	2155s					
3L3	249480	457	**4916s**	6500s					
4K9	99792	145	**2.6h**	3.8h					
$3L5^*$	10020	1001	**4.4h**	>6h					

TABLE 6.1: Computing TMEsCor: For the top to the bottom, these games are harder and harder to be solved. '$> n$h' means that we terminate algorithms after n hours, which also represents that they do not converge within n hours for the remaining cases. To solve large games in our machine, $3L5^*$ has five cards, and team players do not take action 'raising' in $4L3_1$ (6 cards) and $4L3_2$.

EFG	5K11	5K12	5K13	6K7	6K8	6K9	7K7
CMB	**35s**	**58s**	**104s**	**17s**	**64s**	**216s**	**56s**
CMBZ20	2802s	6319s	>3h	2009s	>3h	>3h	>15h

TABLE 6.2: Computing TMEsCor. Here, team players choose actions in information sets reaching by sequence $\emptyset$ and then take action 'calling' in other information sets.

Without loss of generality, we assume that $\sum_{a\in\chi(I_{i,j})} r_i(\sigma_T(i)a) = r_i(\sigma_T(i))$, but $\sum_{a\in\chi(I_{i,j})} w(\sigma_T(i)a, \sigma_{T\setminus\{i\}}) \neq w(\sigma_T)$. We have:

$$
\begin{aligned}
&\textstyle\sum_{a\in\chi(I_{i,j})} w(\sigma_T(i)a, \sigma_{T\setminus\{i\}}) \neq w(\sigma_T)\\
\Rightarrow &\sum_{a\in\chi(I_{i,j})} r_i(\sigma_T(i)a) \prod_{j\in T\setminus\{i\}} r_j(\sigma_T(j)) \neq r_i(\sigma_T(i)) \prod_{j\in T\setminus\{i\}} r_j(\sigma_T(j))\\
\Rightarrow &\textstyle\sum_{a\in\chi(I_{i,j})} r_i(\sigma_T(i)a) \neq r_i(\sigma_T(i))
\end{aligned}
$$

which causes a contradiction, concluding the proof. □

Corollary 6.1. *Given r_n, the optimal solution of Problem (6.5) is the best response against r_n.*

Proof. This is obtained from Theorems 6.6 and 6.7. □

Therefore, our BR can be represented by Problem (6.5).

ϵ	0.1	0.08	0.06	0.04	0.02	0.01	0.008
CMB	**0.41s**	**0.41s**	**0.41s**	**0.50s**	**0.50s**	**0.58s**	**0.58s**
FTP	0.55s	0.71s	0.82s	1.3s	3.8s	8.0s	11.0s
CMB	**0.08s**	**0.08s**	**0.16s**	**0.16s**	**0.29s**	**0.50s**	**0.87s**
FTP	4.0s	6.6s	8.1s	15.5s	72.3s	181s	243s
CMB	**0.23s**	**0.23s**	**0.31s**	**0.31s**	**0.64s**	**1.1s**	**1.1s**
FTP	34.6s	34.6s	67.7s	94.9s	171s	382s	458s
CMB	**2s**	**3s**	**6s**	**13s**	**41s**	**87s**	**111s**
FTP	228s	307s	458s	689s	1574s	4882s	>5h
CMB	**5s**	**7s**	**13s**	**28s**	**93s**	**745s**	**1533s**
FTP	188s	251s	362s	619s	>5h		
CMB	**23s**	**27s**	**37s**	**108s**	**490s**	**3357s**	**8221s**
FTP	164s	215s	371s	920s	>5h		

TABLE 6.3: Computing $\epsilon\Delta_u$-TMEsCor: Games from top to bottom are 3K4, 3K8, 3K12, 3L3, 3L4, and 3L5, respectively, $\Delta_u = 6$ for $3Kr$ and $\Delta_u = 21$ for $3Lr$ ($|L| \approx 10^6$ with $|\Sigma_i| = 801$ for 3L4, and $|L| \approx 3 \cdot 10^6$ with $|\Sigma_i| = 1241$ for 3L5).

6.2 Experimental Evaluation

We conduct experiments on standard games to evaluate our approach. LPs are solved by CPLEX (version 12.9). All algorithms are performed on a machine with 6-core 3.6GHz CPU and 32GB memory (about 27GB are available).

The standard games that we use are the multiplayer Kuhn poker and Leduc Hold'em poker games (see their rules in Farina et al. (2018)), where nKr denotes an n-player Kuhn instance with r ranks (i.e., r cards), and nLr denotes an n-player Leduc Hold'em instance with r ranks (i.e., $3r$ cards). Δ_u is the difference between the maximum and minimum possible utility of the team. In addition, $|L|$ is the number of terminal nodes, and $|\Sigma_i|$ is the number of sequences per player. Without loss of generality, the adversary is player n.

We test our approach on runtimes. To evaluate CMB for computing exact TMEsCor, we have three prior state-of-the-art baselines: 1) C18: a prior algorithm [15] discussed in Section 2.3.4; 2) F18: CG uses BR proposed by Farina et al. (2018) discussed in Section 2.3.4; and 3) CMBZ20: CMB uses the prior algorithm [16] discussed in Sections 2.3.4 and 6.1.3 to generate associated constraints. For ablations, we choose: 1) CMB/A: CMB does not use associated constraints; 2) CMB/ART: CMB does not use the ART and our BR formulation but uses continuous variables to represent reaching

EFG		3K8	3K9	3K10	3K11	3K12
Runtime	TMECor	**4s**	**4s**	**5s**	**10s**	**10s**
	TME	4s	5s	84s	437s	118s
Utility	TMECor	**-0.01928**	**-0.01786**	**-0.01569**	**-0.01456**	**-0.01401**
	TME	-0.06580	-0.04383	-0.06767	-0.05037	-0.05453
	Gap	71%	59%	77%	71%	74%

TABLE 6.4: The runtimes to compute TMEs and TMEsCor and the team's utilities of the team in TMEs and TMEsCor. The gap is the relative distance between the team's utility (V_{Cor}) in a TMECor and the one (V_{TME}) in a TME, i.e., $\frac{|V_{TME}-V_{Cor}|}{|V_{TME}|} \times 100\%$, which means that the team will lose more if the lap is larger.

probabilities for terminal nodes; and 3) CMB/H (respectively, CMB/ART/H): BR in CMB (respectively, CMB/ART) generates pure NFSs for all team members instead of an HFS. Results in Tables 6.1 and 6.2 (CMB and CMBZ20 have similar results in Table 6.1 (results for CMBZ20 are not shown) and then are further evaluated in larger games in Table 6.2) show that: 1) CMB is several orders of magnitude faster than baselines in large games, whose gaps increase with the sizes of games; and 2) CMB significantly outperforms ablation algorithms in large games, which means that each component in CMB significantly enhances the performance.

To evaluate CMB for computing $\epsilon\Delta_u$-TMEsCor, we have a baseline: Fictitious Team Play (FTP) [28] discussed at the beginning of this chapter. Results in Table 6.3 show that CMB runs significantly faster than FTP and is at least two orders of magnitude faster than FTP in large games with small ϵ (e.g., $\epsilon = 0.01$). Our experimental results also show that it is almost impossible for FTP to converge to an $\epsilon\Delta_u$-TMECor with very small ϵ (e.g., $\epsilon = 0.0001$), let alone an exact TMECor. For example, for 3K4, the smallest game in our experiments, FTP cannot converge to an $\epsilon\Delta_u$-TMECor with $\epsilon = 0.0004$ within 100 hours (ϵ reaches 0.0005 within 2674s but then fluctuates around 0.001). Conversely, our CMB, even computing an exact TMECor, only needs 0.7s shown in Table 6.1.

It has been shown [28] that the team will lose a large utility if playing the strategy in TMEs instead of the one TMEsCor in games 3K3–3K7. Now we show runtimes to compute TMEs and TMEsCor and the team's utilities in TMEs and TMEsCor in 3K8–3K12 in Table 6.4. We use the state-of-the-art algorithm [16] to compute $\epsilon\Delta_u$-TMEs. Here, $\epsilon = 0.01$ due to the difficulty to compute an exact TME, and the team's utility is

EFG	3K4	3K6	3K8	3K10	3K12	3L3
Iterations	14	36	47	45	52	1022
Support size	3	5	8	6	10	63

TABLE 6.5: The number of iterations when CMB converges and the size of the support set of the team's strategy in the corresponding TMECor.

obtained when the team's strategy in an $\epsilon\Delta_u$-TME against the adversary's best response. We can see that we need dramatically more time to compute a TME than to compute a TMECor, where the team's utility in a TME is significantly lower than the one in a TMECor.

Table 6.5 shows the number of iterations when CMB converges and the size of the support set of the team's strategy in the corresponding TMECor. We can see that the number of iterations is significantly smaller than the theoretical upper bound in Theorem 6.2 that is larger than $2^{33} \times 33$ (33 is Σ_i in 3K4, where Π_i is significantly larger than Σ_i). In the future, we can explore the tighter theoretical upper bound for the number of iterations. We also can see that the support set in a TMECor is very small.

6.3 Chapter Summary

In this chapter, we propose an efficient algorithm to compute TMEsCor for zero-sum multiplayer EFGs. We first propose our novel HFS representation for the team. We then develop our novel CMB including a novel multilinear BR. In addition, we develop our novel ART to solve our multilinear BR. Finally, experimental results show that CMB is orders of magnitude faster than baselines in large games.

Chapter 7

Computing Team-Maxmin Equilibria with Communication Device for Optimal Interdiction of Urban Criminals with the Aid of Real-Time Information

This chapter[1] focuses on computing TMEsCom [15] for optimal interdiction of urban criminals with the aid of real-time information. This chapter uses the TMECom as the solution concept for the novel network pursuit game model that captures the interaction between an escaping adversary and a team of multiple defense resources (defender) with real-time information available on urban networks. To incorporate such real-time information, we model the urban network security problem as a zero-sum *NEtwork pur-SuiT game (NEST)*. In NEST, the adversary picks one escape path to the outside world, while the defender, tracking the adversary's previous moves, decides on the relocation of her security resources (modeled as a finite-horizon Markov Decision Process (MD-P)). After modeling NEST as an imperfect-recall game to reduce the strategy space,

[1]The work in this chapter has been published as **Youzhi Zhang**, Qingyu Guo, Bo An, Long Tran-Thanh, Nicholas Jennings. Optimal interdiction of urban criminals with the aid of real-time information. *Proceedings of the 33rd AAAI Conference on Artificial Intelligence* (**AAAI**), pp.1262-1269, 2019.

we fist show that the ignorance of real-time information in existing work can lead to an arbitrarily large loss of efficiency. Second, we show that solving NEST is NP-hard. Third, after transforming the non-convex program of solving NEST to a linear program (*LP*), we propose our *ISG* with the following novelty: (i) novel pruning techniques in our *BR*, including: (1) providing tight lower and upper bounds by exploiting the combinatorial structure of each state's action space; and (2) handling imperfect recall by developing an effective lower bound transition method that causes an evaluated state to be reevaluated only if a prefix state with a smaller lower bound transits to it; and (ii) novel techniques to speed up our *ISG* to solve extremely large-scale problems, which include: (1) mapping the computed defender strategy by *BR* between similar subgames to speed up *BR*, which also speeds up *LP* if this strategy is optimal; and (2) adding multiple best response strategies to the restricted game at one iteration against different sampled adversary strategies to reduce the number of iterations for convergence. Finally, experiments show that our approach can scale up to problem sizes with hundreds of nodes on networks including the real network of Manhattan. Thus, for the first time, this chapter shows that *ISG* techniques can also be scaled for dynamic games when best-response oracles are tuned with domain-specific pruning techniques, and other domain-specific improvements are employed.

7.1 Motivating Scenario

To illustrate the underlying issues and motivation for our model, we analyze the problem of police vehicle pursuits. Recently, a new pursuit technology [32, 82] based on GPS has been tested and deployed in many cities, e.g., New York [83] in America and Delta [84] in Canada. This technology developed by StarChase (see *http://www.starchase.com*) provides police officers with real-time (every 3 to 5 seconds [32]) GPS locations of the fleeing vehicle. However, it does not mean that the evader will always be captured. For example, in the urban scenario discussed in [32], even obtaining the real-time information provided by StarChase GPS-based system, police officers only achieved 61% apprehension rate. Outside the cities, i.e., in the open area, it may be even harder to capture the evader. Cities, however, have more advanced facilities and security resources, which make it much easier to locate the evader's location in real time and capture him. Therefore, we can set the roads to the outside of cities as the exit points. In addition, based on the data in [85], about 10% of 5,568 pursuits ended because the vehicle crossed into another jurisdiction. In such a context, we can set the roads to other jurisdictions as the exit points. Meanwhile, about 50% of cases discussed in [82] lasted more than 5 minutes, and the maximum time is 50 minutes. Therefore, police officers have limited but sufficient time to deploy security resources to interdict the evader with the aid of the real-time information about the evader.

Our approach is also suitable for security problems in other domains including interdicting poachers in the field, where UAVs can provide poachers' location information in real time [86]; and combating the threat of piracy or fighting against illegal fishing on the sea, where a new system can track ships or boats in real time [87].

7.2 Problem Description

The *NEtwork purSuiT game (NEST)* is played between an evader (adversary) and police officers (defender) on an urban road network $G = (V, E)$ consisting of a set of directed edges E representing roads and a set of nodes V representing intersections. W.l.o.g., we assume that the time it takes to travel through each road segment is one time unit, e.g., 1 minute, since we can add dummy nodes to separate each road into several segments with equal length. We denote by V_e the set of exit nodes, through which the adversary can escape to the external world. The adversary, initially located at node v_0^a, tries to escape to the external world, while the defender deploys m identical resources (police officers/teams), initially located at intersections $v_0^1, \ldots, v_0^m \in V$, to catch him. The adversary is caught if he and one of the defender resources reach the same node at the same time[1]. We assume that $v_0^a \neq v_0^r$ for all $r \in \mathcal{R} = \{1, \ldots, m\}$. The adversary cannot see the defender until he gets caught [20, 22]. Motivated by the fact that new pursuit technology can be used to provide real-time information about the adversary's whereabouts, the defender can observe the adversary's locations in real time. The initial

[1]Our model can be easily extended to cover the case that both players meet on an edge by extending this definition of capture.

positions of both players are common knowledge. For example, police officers are notified about the location of the crime, while the locations of the police stations are public information.

Adversary's Strategies: A pure adversary strategy is a path from his initial node v_0^a to an exit node $v_e \in V_e$, represented by a sequence of nodes o visited by the adversary with a length no longer than $t_{\max}$[1] (time horizon), where any pair of consecutive nodes are connected by an edge in E. The set of adversary pure strategies is denoted by O. The mixed strategy is a probability distribution over O, denoted by $\mathbf{y} = \langle y_o \rangle$ where y_o represents the probability of escaping through path o. An observed history h is a sequence of nodes where the adversary is spotted. We say that h' is a descendant of h or $h \sqsubset h'$ if h is any strict prefix of h'. We denote by $O_h = \{o \mid o \in O, h \sqsubseteq o\}$ the set of paths generating history h, $H = \{h | \exists o \in O, h \sqsubseteq o\}$ the set of valid histories, and $CH_h = \{h' \mid h' \in H, \exists v \in V, h' = h \cdot v : v$ will be reached at the next step after generating history $h\}$ the set of child histories of h.

Defender's Strategies: The defender's decision problem is modeled as an MDP. Let $l = (v_l^1, ..., v_l^m)$ denote the location of m resources, where v_l^r denotes the location of resource r. Let L represent the set of all possible locations. We say two locations l and l' are adjacent if l' can be reached from l with a one-step move. That is, for any $r \in R$, either $v_l^r = v_{l'}^r$ or $(v_l^r, v_{l'}^r) \in E$. Let $l_0 = (v_0^1, \ldots, v_0^m)$ be the initial location of m resources. A pure strategy for the defender is a deterministic policy $\pi : S \to A$. Each state $s \in S$ consists of two components, the current location $l_s \in L$ and the observed history $h_s \in H$ of the adversary's moves from the initial state to state s, i.e., $s = (l_s, h_s)$. Let $\eta : H \to V$ return the last node in history $h \in H$. A state s is called a capture state if there exists a resource $r \in R$ such that $v_{l_s}^r = \eta(h_s)$. We denote by S_c the set of capture states. On the other hand, if $\eta(h_s) \in V_e$ and there exists no resource allocated on $\eta(h_s)$, the adversary successfully escapes, and such a state is called an escape state. Let S_e denote the set of all escape states. Let the terminal state set be $S_t = S_c \cup S_e$ and the initial state be $s_0 = (l_0, \langle v_0^a \rangle)$. Each action $a \in A$ corresponds to a one-step move from the current location l to the adjacent location l'. Thus, we abuse the notation slightly and

[1]Assume that after $t_{\max}$, the defender can deploy sufficient police forces (e.g., police from neighbor cities) to prevent the adversary from escaping.

let A and L be exchangeable, such that each action is a location to which the defender plans to transfer her resources. We denote by A_s the set of actions in $s \in S \setminus S_t$. Specifically, we denote by $S_{s,l}$ the set of states reached from s by taking action $l \in A_s$, i.e., $S_{s,l} = \{s' \mid s' = (l, h), h \in CH_{h_s}\}$, and we say that $s' \in S_{s,l}$ is a succeeding state of s. We denote by $\mathbf{x}$ the behavior strategy of the defender, where in each non-terminal state s, $\mathbf{x}$ defines a probability distribution over A_s.

Utilities: Assume that the game is zero-sum as we are only concerned with whether the defender can capture the adversary or not. If it is a capture state, the defender receives a unit reward, and the adversary suffers a unit loss; otherwise, both players receive a zero payoff. That is, $u_d(s) = -u_a(s) = 1$ if $s \in S_c$ and $u_d(s) = u_a(s) = 0$ if $s \in S_e$. Given strategy profile $(\mathbf{x}, o)$, we denote by $P_{\mathbf{x},o}(s)$ the probability that state s is reached. Notice that: i) if $h_s \not\sqsubseteq o$, i.e., the escaping path o does not generate history h_s, $P_{\mathbf{x},o}(s) = 0$; and ii) otherwise, $P_{\mathbf{x},o}(s)$ only depends on $\mathbf{x}$ and h_s, and is independent of the adversary's moves after h_s. Therefore, we can denote by $P_{\mathbf{x}}(s)$ the probability of reaching state s under $(\mathbf{x}, o)$ with $h_s \sqsubseteq o$. $P_{\mathbf{x}}(s)$ is determined by the following recursive equation:

$$\begin{aligned} &P_{\mathbf{x}}(s_0) = 1, \\ &P_{\mathbf{x}}(s) = \textstyle\sum_{s' \in S \setminus S_t : s \in S_{s', l_s}} P_{\mathbf{x}}(s') x_{s', l_s} \;\; \forall s \in S \setminus \{s_0\}, \end{aligned} \tag{7.1}$$

where the summation is over states that can transit to s.

Given profile $(\mathbf{x}, o)$, the defender's expected utility is: $U_d(\mathbf{x}, o) = \sum_{s \in S_t : h_s \sqsubseteq o} P_{\mathbf{x}}(s) u_d(s) = \sum_{s \in S_c : h_s \sqsubseteq o} P_{\mathbf{x}}(s)$. Accordingly, we have $U_d(\mathbf{x}, \mathbf{y}) = \sum_{o \in O} y_o U_d(\mathbf{x}, o)$. Since the game is zero-sum, $U_a(\mathbf{x}, \mathbf{y}) = -U_d(\mathbf{x}, \mathbf{y})$.

Equilibrium: We use the NE, (i.e., TMECom) as the solution concept. Generally, $(\mathbf{x}^\star, \mathbf{y}^\star)$ is NE if and only if: (i) $U_d(\mathbf{x}^\star, \mathbf{y}^\star) \geq U_d(\mathbf{x}, \mathbf{y}^\star), \forall \mathbf{x}$; and (ii) $U_a(\mathbf{x}^\star, \mathbf{y}^\star) \geq U_a(\mathbf{x}^\star, \mathbf{y}), \forall \mathbf{y}$.

7.3 Theoretical Analysis

This section first shows the cost of ignoring the real-time information, and then proves that solving NEST is NP-hard.

Cost of Ignoring Information: Here, we denote the strategy that does not consider the up to date information about the adversary as the non-real-time strategy and its counterpart as the real-time strategy.

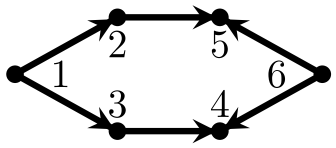


FIGURE 7.1: Example

Consider the example in the left figure, where a unique defender resource is at node 6, the adversary is at node 1, and exit nodes are 4 and 5. If the defender ignores the real-time information, in the worst case (under the NE of the zero-sum game), the adversary reaches each exit node with probability 0.5, while the defender's optimal non-real-time strategy moves the single resource to nodes 4 or 5 with equal probability. That is to say, the defender's expected utility is 0.5. However, the defender can ensure the capture of the adversary by doing the following. (i) If the adversary's current location is at the starting node, the defender just stays at node 6. (ii) When the adversary moves to node 2, the defender moves her resource to node 5, and 4 if she observes that the adversary has moved to node 3. As long as the time horizon is finite, the defender will capture the adversary for sure, and the expected utility is 1, much higher than 0.5.

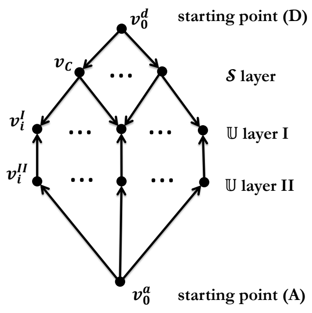
Proposition 7.1. *The defender's optimal non-real-time strategy under the NE can be arbitrarily worse compared with the optimal real-time strategy under the NE.*

Proof. We extend the example in Figure 7.1 to a more general form. That is, there are $|O| = n$ different paths from the adversary's initial location to the external world through n different exit nodes with two time steps. Specifically, any two paths do not intersect. On the other hand, there are n different paths starting from the defender resource's initial location v_0 to different exit nodes by one time step. Similar to the toy example in Figure 7.1, the probability of catching the adversary is $U_d = \frac{1}{n}$ by the optimal non-real-time strategy under the NE, while the probability of catching the adversary is $U_d^\star = 1$ by the optimal real-time strategy under the NE. Then, $\frac{U_d^\star}{U_d} = n$, can be made arbitrarily large by increasing n. □

Complexity Analysis

Theorem 7.1. *Computing an NE of NEST is NP-hard.*

Proof. We reduce the Set Cover problem to computing an NE of NEST, which is described as follows: given a set $\mathbb{U}$ of elements, a collection $\mathcal{S} \subseteq 2^{\mathbb{U}}$ of subsets of $\mathbb{U}$, and an integer m, determine whether there exists a set $\mathcal{C} \subseteq \mathcal{S}$ of size m or less, such that $\cup_{C \in \mathcal{C}} = \mathbb{U}$.



Reduction: The network structure in NEST is demonstrated by the right figure, where v_0^a and v_0^d are starting points for the adversary and the defender respectively. Between them, there are three layers of nodes. The $\mathcal{S}$ layer is fully connected to v_0^d where each node v_C corresponds to the set C in $\mathcal{S}$. $\mathbb{U}$ layers I and II are two identical layers representing all the elements in the ground set $\mathbb{U}$. These two layers are connected with each other in an element-wise manner. Each node v_i^I in layer I is linked with v_C in the $\mathcal{S}$ layer if $i \in C$. On the other hand, the adversary can move from v_0^a to any node in the layer II. Moreover, the defender has m resources, all located at v_0^d initially. The time horizon $t_{\max}$ is set to 2, and the nodes in $\mathbb{U}$ layer I are characterized as exit nodes. It is easy to verify that this is a polynomial-time reduction.

With a horizon of two time steps, suppose that the adversary reaches v_i^{II} at the first time step, and the defender is aware of exit node v_i^I chosen by the adversary. If any one of the defender's resources reaches v_C in the $\mathcal{S}$ layer with $i \in C$ at the first time step, the adversary will be captured for sure. On the other hand, if there is no set cover, then it is easy to show that there is a positive probability that the adversary will move to a v_i^{II} such that the corresponding v_i^I is not protected by any defender resources and thus, the adversary can escape. Thus, an NE of NEST answers the Set Cover problem: there exists a set $\mathcal{C} \subseteq \mathcal{S}$ of size m or less covering the ground set $\mathbb{U} \Leftrightarrow$ in an NE, the defender captures the adversary with probability 1, which requires the defender to move the m resources to nodes at $\mathcal{S}$ layer which fully protect $\mathbb{U}$ layer I. □

7.4 Computing the Optimal Solution

This section first transforms the non-convex program of computing an NE to *LP*, then develops our *ISG* algorithm with a novel *BR* oracle, and finally proposes two additional techniques to speed up our *ISG* significantly.

7.4.1 Transformation to the Linear Program

An NE strategy $\mathbf{x}^\star$ of the defender in NEST can be computed by: $\mathbf{x}^\star \in \arg\max_{\mathbf{x}} \min_{o\in O} U_d(\mathbf{x}, o)$, i.e.,

$$\begin{aligned}
\max_{\mathbf{x}} \quad & U_d^\star && \text{(7.2a)}\\
\text{s.t.} \quad & U_d^\star \leq U_d(\mathbf{x}, o), \quad \forall o \in O && \text{(7.2b)}\\
& \textstyle\sum_{l\in A_s} x_{s,l} = 1, \quad \forall s \in S \setminus S_t && \text{(7.2c)}\\
& x_{s,l} \in [0,1], \quad \forall l \in A_s, \forall s \in S \setminus S_t && \text{(7.2d)}
\end{aligned}$$

Unfortunately, program (7.2) is non-convex, since the probability $P_{\mathbf{x}}(s)$ of reaching state s involves multiplication of various entities in $\mathbf{x}$ due to the recursive property defined in Eq.(7.1). To transform it to an *LP*, based on the propagation of the probability starting from s_0 towards terminal states, we develop a flow representation of the defender's strategy.

Definition 7.1. $\mathbf{f}$ *is the flow representation of the defender's strategy where* $f_{s,l}$ *specifies the probability that the defender reaches state* $s \in S \setminus S_t$ *and takes the action* $l \in A_s$.

Given $\mathbf{f}$ and o, the defender's expected utility is: $U_d(\mathbf{f}, o) = \sum_{s\in S_c: h_s \sqsubseteq o} \sum_{s'\in S\setminus S_t: s\in S_{s',l_s}} f_{s',l_s}$, where,

$$\begin{aligned}
\textstyle\sum_{l\in A_{s_0}} f_{s_0,l} &= 1 && \text{(7.3a)}\\
\textstyle\sum_{s'\in S\setminus S_t: s\in S_{s',l_s}} f_{s',l_s} &= \textstyle\sum_{l\in A_s} f_{s,l}, \forall s \in S \setminus (\{s_0\} \cup S_t) && \text{(7.3b)}\\
f_{s,l} &\geq 0, \quad \forall l \in A_s, \forall s \in S \setminus S_t. && \text{(7.3c)}
\end{aligned}$$

Eqs.(7.3a)–(7.3c) ensure that $\mathbf{f}$ is feasible, which is initialized by Eq.(7.3a), and Eq.(7.3b) is the flow conservation equality.

We now show how to derive the associated flow representation from the behavior strategy and vice versa. Given $\mathbf{x}$, its corresponding flow representation $\mathbf{f}$ is:

$$f_{s,l} = P_{\mathbf{x}}(s)x_{s,l}, \quad \forall l \in A_s, \forall s \in S \setminus S_t. \tag{7.4}$$

Given $\mathbf{f}$, $P_{\mathbf{x}}(s){=}\sum_{s' \in S\setminus S_t : s \in S_{s',l_s}} f_{s',l_s}$ if $s{\neq}s_0$, $P_{\mathbf{x}}(s){=}1$ if $s{=}s_0$, and its corresponding behavior strategy $\mathbf{x}$ is:

$$x_{s,l} {=} \begin{cases} \dfrac{f_{s,l}}{P_{\mathbf{x}}(s)} & \text{if } P_{\mathbf{x}}(s) > 0, \\ \dfrac{1}{|A_s|} & \text{otherwise,} \end{cases} \quad \forall l \in A_s, \forall s \in S \setminus S_t. \tag{7.5}$$

We prove that the flow representation is equivalent with the behavior strategy in terms of expected utility in Theorem 7.2.

Theorem 7.2. *For any pair of* $\mathbf{x}$ *and* $\mathbf{f}$ *satisfying Eqs.(7.4) and (7.5),* $U_d(\mathbf{x}, o) = U_d(\mathbf{f}, o)\ \forall o \in O$.

Proof. Given $\mathbf{x}$, we first show that conditions (7.3a)-(7.3c) are satisfied. By Eq.(7.4), we have:

$$\textstyle\sum_{l \in A_{s_0}} f_{s_0,l} = \sum_{l \in A_{s_0}} P_{\mathbf{x}}(s_0)x_{s_0,l} = \sum_{l \in A_{s_0}} x_{s_0,l} = 1,$$

which is Eq.(7.3a). By Eq.(7.2c), $P_{\mathbf{x}}(s) = \sum_{l \in A_s} x_{s,l}P_{\mathbf{x}}(s)$ $(\forall s \in S \setminus S_t)$. Moreover, by Eq.(7.1), $\forall s \in S \setminus (\{s_0\} \cup S_t)$,

$$\textstyle\sum_{s' \in S\setminus S_t : s \in S_{s',l_s}} P_{\mathbf{x}}(s')x_{s',l_s} = \sum_{l \in A_s} P_{\mathbf{x}}(s)x_{s,l}.$$

Further, by Eq.(7.4), $\forall s \in S \setminus (\{s_0\} \cup S_t)$,

$$\textstyle\sum_{s' \in S\setminus S_t : s \in S_{s',l_s}} f_{s',l_s} = \sum_{l \in A_s} f_{s,l},$$

which is Eq.(7.3b). Obviously, Eq.(7.3c) is obtained from Eqs.(7.1), (7.2d) and (7.4). For each $o \in O$,

$$
\begin{aligned}
U_d(\mathbf{x}, o) &= \textstyle\sum_{s \in S_c : h_s \sqsubseteq o} P_{\mathbf{x}}(s) \\
&= \textstyle\sum_{s \in S_c : h_s \sqsubseteq o} \sum_{s' \in S \setminus S_t : s \in S_{s', l_s}} P_{\mathbf{x}}(s') x_{s', l_s} \\
&= \textstyle\sum_{s \in S_c : h_s \sqsubseteq o} \sum_{s' \in S \setminus S_t : s \in S_{s', l_s}} f_{s', l_s}, \\
&= U_d(\mathbf{f}, o).
\end{aligned}
$$

Therefore, $\forall \mathbf{x}$, $\exists \mathbf{f}$ defined by Eq.(7.4) such that $U_d(\mathbf{x}, o) = U_d(\mathbf{f}, o)$ ($\forall o \in O$).

Given $\mathbf{f}$, we define $\mathbf{x}$ in Eq. (7.5). Obviously, $P_{\mathbf{x}}(s) x_{s,l} = f_{s,l}$, and then $U_d(\mathbf{f}, o) = U_d(\mathbf{x}, o)$. Therefore, $\forall \mathbf{f}$, $\exists \mathbf{x}$ defined by Eq. (7.5) such that $U_d(\mathbf{f}, o) = U_d(\mathbf{x}, o)$ ($\forall o \in O$). $\square$

Finally, we obtain the following *LP* equivalent with (7.2).

$$
\begin{aligned}
\max_{\mathbf{f}} \quad & U_d^\star && (7.6a) \\
\text{s.t.} \quad & U_d^\star \leq U_d(\mathbf{f}, o), \quad \forall o \in O && (7.6b) \\
& \text{Eqs.}(7.3a) - (7.3c). && (7.6c)
\end{aligned}
$$

7.4.2 Incremental Strategy Generation

Although we can compute the optimal solution with *LP* (7.6), it does not scale up due to the exponentially large state space. To mitigate this issue, we propose the **I**teratively **G**enerated **R**eachable **S**tates (*IGRS*) algorithm shown in Algorithm 12, which works as follows. Starting from the restricted NEST $G(S', A')$ where $S' \subset S$ and $A' \subset A$, *LP* (7.6) solves the equilibrium $(\mathbf{x}, \mathbf{y})$ on $G(S', A')$ (Line 3).[1] We then solve the defender's best response policy π in the original strategy space (S, A) (Line 4) and expand A' and S' (Lines 5–9).

[1]Note that, at initialization (Line 1), s_0 is temporarily treated as an escape state in order to compute a solution $(\mathbf{x}, \mathbf{y})$.

Algorithm 12: *IGRS*

```
1  Initialize $S' = \{s_0\}$ and $A' = \emptyset$;
2  repeat
3      $(\mathbf{x}, \mathbf{y}) \leftarrow$ solution of Problem (7.6) under $(S', A')$;
4      $(V, \pi) \leftarrow BR(s_0, U_d(\mathbf{x}, \mathbf{y}))$;
5      for each $s$ reached from $s_0$ by $\pi$ with $V(s) > 0$ do
6          if $s \notin S'$ then
7              $S' \leftarrow S' \cup \{s\}$, $A' \leftarrow A' \cup \{\pi(s)\}$
8          else
9              if $\pi(s) \notin A'_s$ then $A'_s \leftarrow A'_s \cup \{\pi(s)\}$;
10 until $V(s_0) = U_d(\mathbf{x}, \mathbf{y})$;
11 return $(\mathbf{x}, \mathbf{y})$.
```

Defender's Best Response: Our **B**est-**R**esponse (*BR*) algorithm shown in Algorithm 13 computes the best deterministic π against the adversary's strategy $\mathbf{y}$. Due to the large strategy space, to speed up, *BR* adopts a branch-and-bound approach in a depth-first manner. A branch is a state with an expected value representing the probability of catching the adversary starting from this branch. The lower bound of a branch is the minimum value that this branch must obtain in order to be part of the best response, while its upper bound is the maximum value. Using these bounds, *BR* cuts branches which will certainly not be part of the best response.

BR works as follows. Given a state s, if *BR* does not terminate at Lines 1 and 2, *BR* estimates bounds and cuts branches if: (1) the upper bound B_l of action l is less than its lower bound $(V(s) + \epsilon)$ (Line 13); (2) the upper bound $B_{(l,h)}$ of succeeding state (l, h) is not more than 0 (Line 15) or is less than its lower bound $b_{(l,h)}$ (Line 17); or (3) the state value $V(l, h)$ is less than $b_{(l,h)}$ (Line 20). Then, *BR* updates action value Q_l and returns state value $V(s) = \max_l Q_l$ (Lines 21–24).

Specifically, *BR* estimates tight bounds as follows. Firstly, given the combinatorial action space, *BR* estimates tight upper bounds (Lines 3–10 and 14). The idea is that, given a joint action l, *BR* can estimate all the possible paths that the defender has a chance to interdict (i.e., the possibly interdicted path set (PIPS)) by taking l. That is, given a resource r's action l_r in l and an adversary path $o \in O_{h_s}$, *BR* can determine if r taking l_r has a chance to interdict o by comparing the time from its current adversary location $\eta(h_s)$ to exit node $v_e \in o$ through o and the shortest time from action (location)

Algorithm 13: *BR*(s,b_s): $s-$current state, b_s-lower bound

```
1  if $s$ is visited and $b_s \geq b(s)$ then return $V(s)$;
2  if $s \in S_t$ then $V(s) \leftarrow u_d(s) \times \sum_{o \in O_{h_s}} y_o$, return $V(s)$;
3  for $r \in \mathcal{R}$ do
4      for $l_r \in A_s(r)$,$o \in O_{h_s}(y_o > 0)$ do
5          if $dist(l_r, V_e \cap o) + |h_s| + 1 \leq |o|$ then
6              $O_{r,l_r} \leftarrow O_{r,l_r} \cup \{o\}$, $A'_s(r) \leftarrow A'_s(r) \cup \{l_r\}$;
7      if $A'_s(r) = \emptyset$ then $A'_s(r) \leftarrow \{v^r_{l_s}\}$;
8  $A'_s \leftarrow \times_{r \in \mathcal{R}} A'_s(r)$: the best joint action candidate set;
9  for $l \in A'_s$ do $O_l \leftarrow \cup_{r \in \mathcal{R}} O_{r,l_r}$;
10 sort $l \in A'_s$ descending according to value $B_l \leftarrow \sum_{o \in O_l} y_o$;
11 $V(s) \leftarrow \max\{b_s - 2 \times \epsilon, 0\}$, $b(s) \leftarrow -\infty$, $Q_l \leftarrow 0 (\forall l \in A'_s)$;
12 for $l \in A'_s$ do
13     if $V(s) + \epsilon \leq B_l$ then
14         sort $(l,h)$ $(h \in CH_{h_s})$ descending according to value $B_{(l,h)} \leftarrow \sum_{o \in O_h \cap O_l} y_o$,
             $B' \leftarrow \sum_{h \in CH_{h_s}} B_{(l,h)}$;
15         for $h \in CH_{h_s}, B_{(l,h)} > 0$ do
16             $b_{(l,h)} \leftarrow V(s) + \epsilon - (Q_l + (B' - B_{(l,h)}))$;
17             if $b_{(l,h)} > B_{(l,h)}$ then break;
18             else
19                 $V(l,h) \leftarrow$BR$((l,h), b_{(l,h)})$;
20                 if $b_{(l,h)} > V(l,h)$ then break;
21                 else $Q_l \leftarrow Q_l + V(l,h)$, $B' \leftarrow B' - B_{(l,h)}$;
22         if $V(s) < Q_l$ then $V(s) \leftarrow Q_l, \pi(s) \leftarrow l$;
23 if $V(s) < b_s$ then $b(s) \leftarrow b_s$;
24 return $V(s)$.
```

l_r to v_e (Line 5). Thus, *BR* obtains l_r's PIPS (O_{r,l_r} (Line 6) initialized as $\emptyset$ with the corresponding best action candidate set $A'_s(r)$), l's PIPS (O_l (Line 9)), and (l,h)'s PIPS ($O_l \cap O_h$ (Line 14)). Secondly, to be the best action, an action's value must be larger than the obtained maximum action value, and then the lower bounds for the remaining actions and succeeding states will adaptively increase. Specifically, $V(s)$ is initialized at Line 11 to guarantee that the best action is computed if s's optimal value $V^\star(s) = b_s$; and $b_{l,h}$ is estimated by assuming that all the remaining succeeding states will return the maximum value (Line 16).

In addition, *BR* effectively handles imperfect recall of NEST where a state may be visited by different prefix states. To avoid repeated computation, *BR* stores the computed state value for state s as $V(s)$ that is immediately returned if s is visited again (Line

1). However, while setting tight bounds, only using this method may cause some wrong cuts. Here, if b_s (that is set for s while being visited by its prefix state s_1) is tight, and $V(s)$ is computed such that $V(s) < b_s$, then $V(s)$ may not be optimal because some cuts may happen. If $V(s)$ is not optimal (i.e., $V(s) < V^\star(s) < b_s$) and s is revisited by another prefix state s_2 with b'_s such that $V(s) < b'_s < V^\star(s)$, then the wrong cut will happen in s_2 after $V(s)$ is returned to s_2. To avoid these wrong cuts, *BR* records the computed state value with the corresponding lower bound (recorded as $b(s)$) if this value is less than this bound (Line 23), and this value will be recomputed only if this state's prefix state with a smaller lower bound transits to it (Line 1).

Theorem 7.3. IGRS *converges to an NE.*

Proof. After calling our *BR* algorithm, we can obtain the best response policy starting from s_0 against $\mathbf{y}$ in $G(S, A)$. Note that $V(s_0)$ is the defender utility by the best response against $\mathbf{y}$ in $G(S, A)$ by our *BR* algorithm. Then $V(s_0) \geq U_d(\mathbf{x}, \mathbf{y}) \geq 0$. If $V(s_0) > U_d(\mathbf{x}, \mathbf{y})$, π, i.e., the output at Line 4, must contain some states s with $V(s) > 0$ or their actions that are not in $G(S', A')$. Consequently, new states and actions will be added to $G(S', A')$, and then $G(S', A')$ is expanded. In the worst case, $G(S', A') = G(S, A)$, where *IGRS* will stop and $V(s_0) = U_d(\mathbf{x}, \mathbf{y})$. Therefore, *IGRS* will converge with $V(s_0) = U_d(\mathbf{x}, \mathbf{y})$ with a finite number of iterations because the number of states and actions in $G(S, A)$ is finite. Therefore, $U_d(\mathbf{x}, \mathbf{y}) \geq U_d(\mathbf{x}', \mathbf{y})$ $(\forall \mathbf{x}')$. Note that $U_a(\mathbf{x}, \mathbf{y}) \geq U_a(\mathbf{x}, \mathbf{y}')$ $(\forall \mathbf{y}')$. Then, $(\mathbf{x}, \mathbf{y})$ is an NE. □

7.4.3 Improving the Scalability of *IGRS*

Even though our *IGRS* is far more efficient than *LP*, it still cannot handle the very large NEST. Thus, we further improve the scalability of *IGRS* by developing two techniques.

7.4.4 Mapping Between Subgames

The first technique is to map the computed defender strategy by *BR* in a subgame to its similar subgames to speed up *BR*. Here, NEST is the full game, and part of NEST is the subgame. Moreover, if this computed strategy is optimal in NEST (i.e., part of the equilibrium in NEST), we can also speed up *LP* and map this strategy to more subgames. We formally define this technique and then illustrate it.

Formally, subgame G_s has initial state s, where the defender's strategy space (S_{G_s}, A_{G_s}) includes all states reachable from s (i.e., if s_1 in S_{G_s} transits to s_2, then $s_2 \in S_{G_s}$) with the corresponding action space, and the adversary's strategy space includes all paths generating h_s, i.e., O_{h_s}.[1] Here, the defender has the same action space in states with the same location for her. Given states s_1 and s_2, $O_{h_{s_1}}$ is similar to $O_{h_{s_2}}$ after $\eta(h_{s_1})$ if $\eta(h_{s_1}) = \eta(h_{s_2})$, and a one-to-one correspondence exists between $O_{h_{s_1}}$ and $O_{h_{s_2}}$ such that each such pair $o_1 \in O_{h_{s_1}}$ and $o_2 \in O_{h_{s_2}}$ satisfies $q(h_{s_1}, o_1) = q(h_{s_2}, o_2)$ ($q(h_1, h_2)$ is a subsequence of h_2 after $\eta(h_1)$ such that $h_1 \cup q(h_1, h_2) = h_2$). That is, the adversary has the same move space after $\eta(h_{s_1})$ by both sets.

Two subgames G_{s_1} and G_{s_2} are similar if $l_{s_1} = l_{s_2}$, and $O_{h_{s_1}}$ is similar to $O_{h_{s_2}}$ after $\eta(h_{s_1})$. Then, states $s_i \in S_{s_1}$ and $s_j \in S_{s_2}$ are similar if $l_{s_i} = l_{s_j}$ and $q(h_{s_1}, h_{s_i}) = q(h_{s_2}, h_{s_j})$. The best response strategy π_{s_1} in G_{s_1} against $\mathbf{y}$ is defined by: Given the best response π against $\mathbf{y}$ by *BR*,

$$\pi_{s_1}(s') = \pi(s')(\forall s' \in S_{G_{s_1}}). \tag{7.7}$$

Here, for $s' \in S_{G_{s_1}}$, if s' is reached from s_1 by π_{s_1}, we define $P_{\pi_{s_1}}(s') = 1$; otherwise, $P_{\pi_{s_1}}(s') = 0$. Therefore, a mapping strategy $\pi_{s_1 \to s_2}$ from G_{s_1} to its similar subgame G_{s_2} is defined by: Given π_{s_1} defined by Eq.(7.7), $\forall s_j \in S_{G_{s_2}}$,

$$\pi_{s_1 \to s_2}(s_j) = \pi_{s_1}(s_i), \tag{7.8}$$

where s_i in $S_{G_{s_1}}$ and s_j are similar. We use this mapping only if the adversary takes both paths at each pair of the one-to-one correspondence between $O_{h_{s_1}}$ and $O_{h_{s_2}}$ with the same probability. This mapping could happen at the iteration when π_{s_1} is computed during calling *BR* or at the future iterations due to the loop in *IGRS*.

[1]We define the subgame like this because we only compute the defender's strategies in subgames through *BR* given $\mathbf{y}$ over O.

Theorem 7.4. *For any G_{s_1} with π_{s_1} defined by Eq.(7.7) as the best response in G_{s_1} against $\mathbf{y}$, if G_{s_2} is its similar subgame with $\mathbf{y}'$ satisfying $y_{o_1} = y'_{o_2}$ for each pair $o_1 \in O_{h_{s_1}}$ and $o_2 \in O_{h_{s_2}}$ with $q(h_{s_1}, o_1) = q(h_{s_2}, o_2)$ in the one-to-one correspondence between $O_{h_{s_1}}$ and $O_{h_{s_2}}$, then $\pi_{s_1 \to s_2}$ defined by Eq.(7.8) is the best response in G_{s_2} against $\mathbf{y}'$.*

Proof. Because $O_{h_{s_1}}$ in G_{s_1} is similar to $O_{h_{s_2}}$ in G_{s_2}, we have

$$\begin{aligned}
V^{\pi_{s_1 \to s_2}}(s_2) &= \sum_{P_{\pi_{s_1 \to s_2}}(s_c)=1, s_c \in S_c \cap S_{G_{s_2}} : h_{s_c} \sqsubseteq o \in O_{h_{s_2}}} y'_o \\
&= \sum_{P_{\pi_{s_1}}(s_c)=1, s_c \in S_c \cap S_{G_{s_1}} : h_{s_c} \sqsubseteq o \in O_{h_{s_1}}} y_o \\
&= V^{\pi_{s_1}}(s_1).
\end{aligned}$$

Suppose $\pi_{s_1 \to s_2}$ is not the best response in G_{s_2} against $\mathbf{y}'$. Then, there is a best response strategy π'_{s_2} against $\mathbf{y}'$ such that $V^{\pi'_{s_2}}(s_2) > V^{\pi_{s_1 \to s_2}}(s_2)$. Let $O_{y'}$ be the support set of $\mathbf{y}'$. If $O_y \cap O_{h_{s_2}} = \emptyset$, then $V^{\pi'_{s_2}}(s_2) = V^{\pi_{s_1 \to s_2}}(s_2) = 0$, which leads to a contradiction. If $O_{y'} \cap O_{h_{s_2}} = O^* \neq \emptyset$, then $V^{\pi'_{s_2 \to s_1}}(s_1) = V^{\pi'_{s_2}}(s_2) > V^{\pi_{s_1 \to s_2}}(s_2) = V^{\pi_{s_1}}(s_1)$, i.e., π_{s_1} is not the best response in G_{s_1} against $\mathbf{y}$, which causes a contradiction. Therefore, $\pi_{s_1 \to s_2}$ is the best response in G_{s_2} against $\mathbf{y}'$. □

A defender strategy $\pi^\star_s$ in G_s is optimal in NEST if given any strategy $\mathbf{y}$ over O and any policy π_s in G_s, $V^{\pi_s}(s) \le V^{\pi^\star_s}(s)$, and we say that this $\pi^\star_s$ is part of the equilibrium in NEST. $\pi^\star_s$ is defined by: Given π_s defined by Eq.(7.7) as the best response in G_s against some $\mathbf{y}$ over O with $y_o > 0$ ($\forall o \in O_{h_s}$) such that $V^{\pi_s}(s) = \sum_{o \in O_{h_s}} y_o$,

$$\pi^\star_s(s') = \pi_s(s') (\forall s' \in S_{G_s}). \tag{7.9}$$

Lemma 7.1. *Given any $\mathbf{y}$ over O, $V^{\pi^\star_s}(s) = \sum_{o \in O_{h_s}} y_o$ under $\pi^\star_s$ defined by Eq.(7.9).*

Proof. If there is a strategy $\mathbf{y}$ such that $V^{\pi^\star_s}(s) = \sum_{o \in O_{h_s}} y_o \sum_{s_c \in S_c \cap S_{G_s} : h_{s_c} \sqsubseteq o} P_{\pi^\star_s}(s_c) < \sum_{o \in O_{h_s}} y_o$ under $\pi^\star_s$, then there is a path $o^* \in O_{h_s}$ such that $\sum_{s_c \in S_c \cap S_{G_s} : h_{s_c} \sqsubseteq o^*} P_{\pi^\star_s}(s_c) = 0$. Therefore, if $\pi^\star_s$ is played against $\mathbf{y}$ with $y_o > 0$ $(\forall o \in O_{h_s})$, $V^{\pi_s^\star}(s) = \sum_{o \in O_{h_s}} y_o \sum_{s_c \in S_c \cap S_{G_s}: h_{s_c} \sqsubseteq o} P_{\pi_s^\star}(s_c) = \sum_{o \in O_{h_s} \setminus \{o^*\}} y_o \sum_{s_c \in S_c \cap S_{G_s}: h_{s_c} \sqsubseteq o} P_{\pi_s^\star}(s_c) < \sum_{o \in O_{h_s}} y_o$, which contradicts the definition of $\pi_s^\star$ in Eq.(7.9). □

Theorem 7.5. *$\pi_s^\star$ defined by Eq.(7.9) is optimal in NEST.*[1]

Proof. Suppose $\pi_s^\star$ is not optimal in NEST. Then, there is a strategy $\mathbf{y}$, and π_s is the best response in G_s such that $V^{\pi_s}(s) > V^{\pi_s^\star}(s)$. Let O_y be $\mathbf{y}$'s support set. If $O_y \cap O_{h_s} = \emptyset$, then $V^{\pi_s}(s) = V^{\pi_s^\star}(s) = 0$, which leads to a contradiction. If $O_y \cap O_{h_s} = O^* \neq \emptyset$, then, by Lemma 7.1, $V^{\pi_s^\star}(s) = \sum_{o \in O_{h_s}} y_o = \sum_{o \in O^*} y_o \geq V^{\pi_s}(s)$, which also causes a contradiction. □

Given $\pi_s^\star$ defined by Eq.(7.9), the defender will certainly catch the adversary in G_s. Therefore, after computing $\pi_s^\star$ in G_s, if G_s is reached again, the defender's expected utility $\sum_{o \in O_{h_s}} y_o$ is returned immediately. Therefore, *LP* for the restricted game can exclude constraints and variables for the succeeding states of s; and, for any adversary strategy, *BR* does not need to compute the best response starting from s anymore, which will speed up *LP* and *BR*.

We can map $\pi_{s_1}^\star$ defined by Eq.(7.9) in G_{s_1} to its similar subgames through Eq.(7.8). Besides, we can map this $\pi_{s_1}^\star$ to more subgames, where the initial location of m resources is not l_{s_1}. In fact, given $\pi_{s_1}^\star$, we can trace the interdiction back to which resource r starting from s_1 contributes to this interdiction by following $\pi_{s_1}^\star$, and we may find that we only need part of m resources to interdict the adversary. Then, we only need to map the actions of these key resources to other subgames to capture the adversary certainly. For example, consider G_{s_1} and G_{s_2}, where $l_{s_1} = (v_1, v_2)$, $l_{s_2} = (v_1, v_3)$, $O_{h_{s_1}}$ is similar to $O_{h_{s_2}}$ after $\eta(h_{s_1})$, and only the first resource in G_{s_1} will contribute to the interdiction by $\pi_{s_1}^\star$. Then, the adversary will be captured certainly in G_{s_2} if the resource initially at v_1 in G_{s_2} takes the action of the resource initially at v_1 in G_{s_1} by $\pi_{s_1}^\star$.

Formally, we define this strategy mapping between semi-similar subgames. Two subgames G_{s_1} and G_{s_2} are semi-similar on $\bar{l}$ if $\bar{l} \subseteq l_{s_1}$, $\bar{l} \subseteq l_{s_2}$, and $O_{h_{s_1}}$ is similar

[1]Note that, for games with imperfect information, the general subgame solving technique cannot guarantee optimality [88].

to $O_{h_{s_2}}$ after $\eta(h_{s_1})$. Similar subgames are certainly semi-similar.[1] Here, $s_i \in S_{s_1}$ and $s_j \in S_{s_2}$ are semi-similar if $\underline{l}$, with $\underline{l} \subseteq l_{s_i}$ and $\underline{l} \subseteq l_{s_j}$, is reached from $\bar{l}$ by the corresponding resources in s_1 and s_2, respectively, and $q(h_{s_1}, h_{s_i}) = q(h_{s_2}, h_{s_j})$.

Given $\pi^\star_{s_1}$ defined by Eq.(7.9), if resource r starting from s_1 by $\pi^\star_{s_1}$ can interdict at least one of the paths in O_{s_1} (i.e., there exists $o \in O_{h_{s_1}}$ such that there is a capture state s_c with $P_{\pi^\star_{s_1}}(s_c) = 1$, $h_{s_c} \sqsubseteq o$, and $v^r_{l_{s_c}} = \eta(h_{s_c})$), we call r's location $v^r_{l_{s_1}}$ in s_1 as the key location. Let $l^*_{s_1} \subseteq l_{s_1}$ be the set of these key locations in s_1. If G_{s_2} and G_{s_1} are semi-similar on $l^*_{s_1}$ with $\pi^\star_{s_1}$ defined by Eq.(7.9) in G_{s_1}, then its mapping strategy $\pi^\star_{s_1\to s_2}$ in G_{s_2} is defined by: $\forall s_j \in S_{G_{s_2}}$,

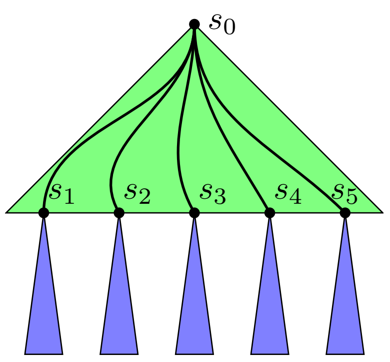
$$\pi^\star_{s_1\to s_2}(s_j) = (\cdots, l_r, \cdots), \tag{7.10}$$

where if $v^r_{l_{s_2}} \in l^*_{s_1}$, then $l_r = \pi^\star_{s_1}(s_i)_{r*, v^{r*}_{l_{s_1}} = v^r_{l_{s_2}}}$ (s_i in $S_{G_{s_1}}$ is semi-similar to s_j with $P_{\pi^\star_{s_1}}(s_i) = 1$) which means that resource r in s_j follows $\pi^\star_{s_1}(s_i)$ for resource $r*$ in s_i whose location $v^{r*}_{l_{s_1}}$ in s_1 overlaps r's location $v^r_{l_{s_2}}$ in s_2; otherwise, $l_r = v^r_{l_{s_j}}$, i.e., staying at the current node.

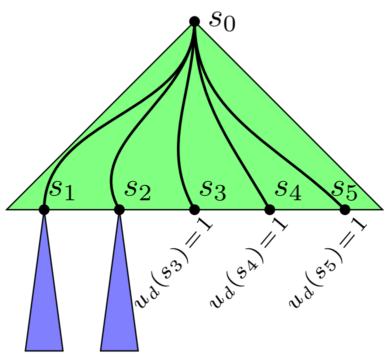
Theorem 7.6. *$\pi^\star_{s_1\to s_2}$ defined by Eq.(7.10) is optimal in NEST.*

Proof. Suppose $\pi^\star_{s_1\to s_2}$ is not optimal in G_{s_2}. Then, there is a strategy $\mathbf{y}$, and π_{s_2} is the best response in G_{s_2} such that $V^{\pi_{s_2}}(s_2) > V^{\pi^\star_{s_1\to s_2}}(s_2)$. Let $O_{\mathbf{y}}$ be $\mathbf{y}$'s support set. If $O_{\mathbf{y}} \cap O_{h_{s_2}} = \emptyset$, then $V^{\pi_{s_2}}(s_2) = V^{\pi^\star_{s_1\to s_2}}(s_2) = 0$, which leads to a contradiction. If $O_{\mathbf{y}} \cap O_{h_{s_2}} = O^* \neq \emptyset$, by the assumption $V^{\pi_{s_2}}(s_2) > V^{\pi^\star_{s_1\to s_2}}(s_2)$, then there is a path $o' \in O_{h_{s_2}}$ which is not interdicted by $\pi^\star_{s_1\to s_2}$, i.e., $P_{\pi^\star_{s_1\to s_2}}(s_c) = 0$ ($\forall s_c \in S_c \cap S_{G_{s_2}}$

[1]We still need to consider similar subgames because we cannot have the property similar to Theorem 7.4 between semi-similar subgames. To illustrate this, we consider the scenario shown in Figure 7.3. For subgame G_{s_1} we know that π_{s_1} ($\pi_{s_1}(s_1) = (6,5)$ transiting to a capture state $s_{11} = ((6,5), h_{11})$ such that $V^{\pi_{s_1}}(s_1) = 0.2$) is the best response in G_{s_1} against the adversary strategy with $y_{o_1} = 0.2$ and $y_{o_2} = 0.1$. Here, only the first resource contributes to the interdiction, i.e., interdicting the adversary in s_{11} due to $\eta(h_{11}) = 6$. Now we consider subgame G_{s_6} with initial state $s_6 = ((11,8), h_2)$. Obviously, G_{s_1} and G_{s_6} are semi-similar on $\bar{l} = (11)$ that is also the key location of G_{s_1}. Let us define the mapping strategy of π_{s_1} as $\pi_{s_1\to s_6}$ by Eq.(7.10). That is, $\pi_{s_1\to s_6}(s_6) = (6,8)$. However, $\pi_{s_1\to s_6}$ is not the best response in G_{s_6} against the adversary strategy with $y_{o_3} = 0.2$ and $y_{o_4} = 0.1$ because taking action $(6,7)$ in s_6 will result in $V(s_6) = 0.3$ (larger than $V^{\pi_{s_1\to s_6}}(s_6) = 0.2$). That is, given a best response (non-optimal) strategy in G_{s_1} with the key location set $l^*_{s_1}$, resources in a semi-similar subgame G_{s_6} of G_{s_1} who are not initially at nodes in $l^*_{s_1}$ may contribute to the interdiction, which results in a better strategy than its mapping strategy. Therefore, we cannot have the property similar to Theorem 7.4 between semi-similar subgames. However, Theorem 7.4 holds between similar subgames.



(a) Before



(b) After using optimal strategies in G_{s_3}, G_{s_4}, and G_{s_5}

FIGURE 7.2: The full game before and after using the optimal strategies in subgames.

with $h_{s_c} \sqsubseteq o'$). Note that, given a deterministic policy π_s in G_s, for each path $o \in O_{h_s}$, there is at most one capture state s_c such that $P_{\pi^\star_s}(s_c) = 1$ and $h_{s_c} \sqsubseteq o$. For each $o_1 \in O_{h_{s_1}}$, if there is a capture state s_c with $P_{\pi^\star_{s_1}}(s_c) = 1$ and $h_{s_c} \sqsubseteq o_1$, then for its similar o_2 in $O_{h_{s_2}}$ such that $q(h_{s_1}, o_1) = q(h_{s_2}, o_2)$, there is a capture state s'_c with $P_{\pi^\star_{s_1\to s_2}}(s'_c) = 1$ and $h_{s'_c} \sqsubseteq o_2$ because resources at the key locations take the same actions in semi-similar states in both subgames by Eq.(7.10). Therefore, $o^* \in O_{h_{s_1}}$ with $q(h_{s_1}, o^*) = q(h_{s_2}, o')$ does not generate the history in any capture state reached from s_1 by $\pi^\star_{s_1}$, i.e., $P_{\pi^\star_{s_1}}(s_c) = 0$ ($\forall s_c \in S_c \cap S_{G_{s_1}}$ with $h_{s_c} \sqsubseteq o^*$). It means that, for $\mathbf{y}'$ with $y'_o > 0 (\forall o \in O_{h_{s_1}})$, $V^{\pi^\star_{s_1}}(s_1) = \sum_{o\in O_{h_{s_1}}} y'_o \sum_{s_c\in S_c\cap S_{G_{s_1}}:h_{s_c}\sqsubseteq o} P_{\pi^\star_{s_1}}(s_c) = \sum_{o\in O_{h_{s_1}}\setminus\{o^*\}} y'_o \sum_{s_c\in S_c\cap S_{G_{s_1}}:h_{s_c}\sqsubseteq o} P_{\pi^\star_{s_1}}(s_c) < \sum_{o\in O_{h_{s_1}}} y'_o$, which contradicts Lemma 7.1 for $\pi^\star_{s_1}$. Therefore, $V^{\pi^\star_{s_1\to s_2}}(s_2) = \sum_{o\in O_{h_{s_2}}} y_o$ for $\mathbf{y}$ with $y_o > 0$ ($\forall o \in O_{h_{s_2}}$), and $\pi^\star_{s_1\to s_2}$ is optimal in NEST by Theorem 7.5. $\square$

Now we illustrate this technique. As shown in Figure 7.2a, in the full game, the defender's strategy space (S, A) includes all states and actions, while the adversary's strategy space O includes all escaping paths. Subgames of G_{s_1}, G_{s_2}, G_{s_3}, G_{s_4}, and G_{s_5} are part of the full game starting from states s_1, s_2, s_3, s_4, and s_5, respectively.

To illustrate the characteristic of subgames, we analyze the scenario shown in the right figure. The adversary arrives at v_{12} through two different directions generating history h_1 $(= \langle v_0^a, v_{13}, v_{12}\rangle)$ and history h_2 $(= \langle v_0^a, v_{14}, v_{12}\rangle)$, respectively. Exit nodes are v_1 and v_2.

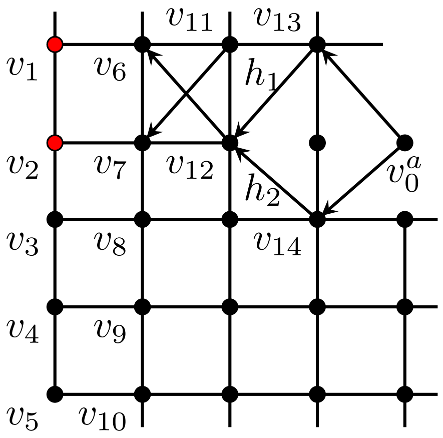


FIGURE 7.3: Scenario

We consider three initial locations for two defender resources: $l_1 = (v_{11}, v_5)$, $l_2 = (v_3, v_4)$, and $l_3 = (v_3, v_5)$. By combining the observed adversary history with the locations of two defender resources, we consider five states $s_1 = ((v_{11}, v_5), h_1)$, $s_2 = ((v_{11}, v_5), h_2)$, $s_3 = ((v_3, v_4), h_1)$, $s_4 = ((v_3, v_4), h_2)$, and $s_5 = ((v_3, v_5), h_2)$, which are the initial states of five subgames: G_{s_1}, G_{s_2}, G_{s_3}, G_{s_4}, and G_{s_5}.

We consider time horizon $t_{\max} = 4$. Then, the set of paths generating h_1 is $O_{h_1} = \{o_1, o_2\}$ with $o_1 = h_1 \cup \langle v_6, v_1\rangle = h_{13}$ and $o_2 = h_1 \cup \langle v_7, v_2\rangle = h_{14}$, while $CH_{h_1} = \{h_{11}, h_{12}\}$ with $h_{11} = h_1 \cdot v_6$ and $h_{12} = h_1 \cdot v_7$, $CH_{h_{11}} = \{h_{13}\}$, and $CH_{h_{12}} = \{h_{14}\}$. The set of paths generating h_2 is $O_{h_2} = \{o_3, o_4\}$ with $o_3 = h_2 \cup \langle v_6, v_1\rangle = h_{23}$ and $o_4 = h_2 \cup \langle v_7, v_2\rangle = h_{24}$, while $CH_{h_2} = \{h_{21}, h_{22}\}$ with $h_{21} = h_2 \cdot v_6$ and $h_{22} = h_2 \cdot v_7$, $CH_{h_{21}} = \{h_{23}\}$, and $CH_{h_{22}} = \{h_{24}\}$. Then, O_{h_1} and O_{h_2} are similar after v_{12} because $\eta(h_1) = \eta(h_2) = v_{12}$, $q(h_1, o_1) = \langle v_6, v_1\rangle = q(h_2, o_3)$, and $q(h_1, o_2) = \langle v_7, v_2\rangle = q(h_2, o_4)$.

O_{h_1} is the adversary's strategy space of G_{s_1} and G_{s_3}, and O_{h_2} is the adversary's strategy space of G_{s_2}, G_{s_4}, and G_{s_5}. Thus, G_{s_1} and G_{s_2} are similar because $(11, 5) = l_{s_1} = l_{s_2}$, and O_{h_1} and O_{h_2} are similar after node $\eta(h_{s_1}) = v_{12}$ $(h_{s_1} = h_1)$. After we compute the best response in G_{s_1} (i.e., π_{s_1} with $\pi_{s_1}(s_1) = (v_6, v_5)$ transiting to a capture state $((6, 5), h_{11})$ such that $V^{\pi_{s_1}}(s_1) = 0.2$) against the adversary strategy $\mathbf{y}$ with $y_{o_1} = 0.2$ and $y_{o_2} = 0.1$, we can map it to G_{s_2} against the adversary strategy with $y_{o_3} = 0.2$ and $y_{o_4} = 0.1$ (Theorem 7.4). That is, the best response strategy for state s_1 is taking $l = (v_6, v_5)$, which is also the best response strategy for its similar state s_2 after the mapping.

Some computed strategies in subgames are the best response against any adversary strategy. The defender strategy in G_{s_3} against the adversary strategy with $y_{o_1} = 0.2$ and $y_{o_2} = 0.1$ is π_{s_3} such that $V^{\pi_{s_3}}(s_3) = 0.3$: $\pi_{s_3}(s_3) = (v_2, v_3)$ transiting to states $s_{31} = ((v_2, v_3), h_{11})$ and $s_{32} = ((v_2, v_3), h_{12})$, $\pi_{s_3}(s_{31}) = (v_1, v_2)$ transiting to capture state $s_{33} = ((v_1, v_2), h_{13})$, and $\pi_{s_3}(s_{32}) = (v_2, v_3)$ transiting to capture state $s_{34} = ((v_2, v_3), h_{14})$. Indeed, by π_{s_3} in G_{s_3}, given any adversary strategy, we have $V^{\pi_{s_3}}(s_3) = y_{o_1} + y_{o_2}$ (Lemma 7.1). That is, starting from s_3, the adversary will be captured certainly. This strategy is denoted as $\pi^{\star}_{s_3}$, which is optimal in NEST (Theorem 7.5). Then, we can define the payoff 1 for the defender in state s_3 as shown in Figure 7.2b, i.e., the defender's expected utility in subgame G_{s_3} is $y_{o_1} + y_{o_2}$. Then, if this subgame is reached again, the defender's expected utility $y_{o_1} + y_{o_2}$ is returned immediately.

$\pi^{\star}_{s_3}$ in G_{s_3} can also be mapped to its semi-similar subgame G_{s_5}. In G_{s_3} and G_{s_5}, the adversary has the same move space and the first defender resource in G_{s_3} and the first defender resource in G_{s_5} has the same initial location (v_3) in their initial states s_3 and s_5, respectively. For strategy $\pi^{\star}_{s_3}$, the first resource will finally catch the adversary in state s_{33} or state s_{34} by moving to v_2 first, then moving to v_1 if she observes the adversary history h_{11} (i.e., in state s_{31}), or staying at v_2 if she observes the adversary history h_{12} (i.e., in state s_{32}). That is, v_3 is the key location in G_{s_3}. Therefore, given the mapping of $\pi^{\star}_{s_3}$ guaranteeing that the first defender resource in G_{s_3} and the first defender resource in G_{s_5} take the same action after observing the adversary's same move in both subgames, this mapping can guarantee optimality (Theorem 7.6). That is, the first defender resource in G_{s_5} will finally catch the adversary in state $s_{53} = ((v_1, v_5), h_{23})$ or state $s_{54} = ((v_2, v_5), h_{24})$ if she moves to v_2 first, then moves to v_1 if she observes the adversary history h_{21} (i.e., in state $s_{51} = ((v_2, v_5), h_{21})$), or stays at v_2 if she observes the adversary history h_{22} (i.e., in state $s_{52} = ((v_2, v_5), h_{22})$) when the first resource does not move. More specifically, the optimal strategy for state $s_{31} = ((v_2, v_3), h_{11})$ (note that $P_{\pi^{\star}_{s_{31}}} = 1$) is taking $l = (v_1, v_2)$, i.e., the first resource in s_{31}, locating at the key location v_2 that is reached from v_3 in s_1, moves to v_1; and then the mapping strategy for its semi-similar state $s_{51} = ((v_2, v_5), h_{21})$ is taking $l' = (v_1, v_5)$, i.e., the first resource in s_{51}, locating at the key location v_2 that is reached from v_3 in s_5, moves to v_1 while the second resource in s_{51} stays at the current node by Eq.(7.10). This $\pi^{\star}_{s_3 \to s_5}$ defined

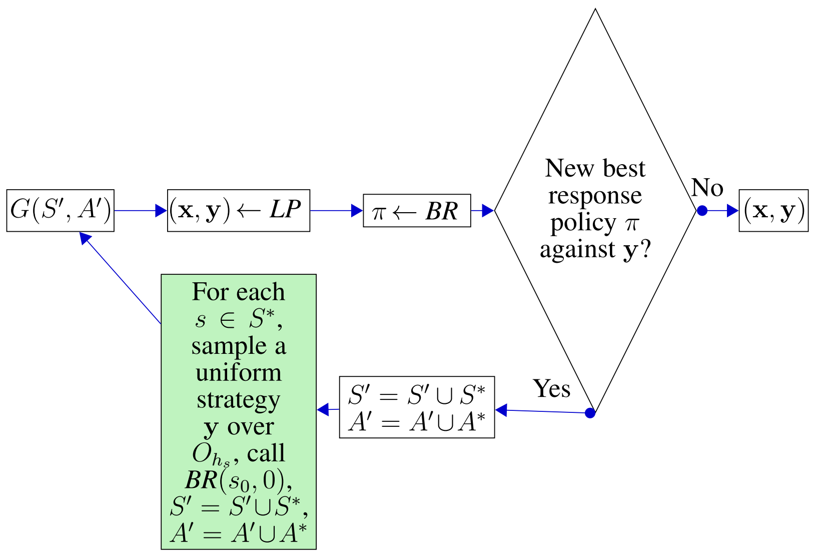

$G(S', A')$
$(\mathbf{x}, \mathbf{y}) \leftarrow LP$
$\pi \leftarrow BR$
New best response policy π against $\mathbf{y}$?
No
$(\mathbf{x}, \mathbf{y})$
Yes
$S' = S' \cup S^*$ $A' = A' \cup A^*$
For each $s \in S^*$, sample a uniform strategy $\mathbf{y}$ over O_{h_s}, call $BR(s_0, 0)$, $S' = S' \cup S^*$, $A' = A' \cup A^*$

FIGURE 7.4: Adding multiple best response strategies at one iteration.

by Eq.(7.10) in G_{s_5} is optimal in the full game. By $\pi^\star_{s_3 \to s_5}$, as shown in Figure 7.2b, we define the payoff 1 for the defender in state s_5, i.e., the defender's expected utility in subgame G_{s_5} is $y_{o_3} + y_{o_4}$. Similarly, G_{s_3} and G_{s_4} are semi-similar and $\pi^\star_{s_3 \to s_4}$ in G_{s_4} by Eq.(7.10) is optimal in NEST.

7.4.5 Adding Multiple Strategies at One Iteration

The second technique is to add multiple defender best response strategies at one iteration to reduce the number of iterations for the convergence of *IGRS*. More specifically, we sample multiple adversary strategies to add the corresponding defender's best response strategies to the restricted game $G(S', A')$ by calling *BR* in *IGRS* at one iteration.

We sample a uniform strategy over O_{h_s} for each state s in $G(S', A')$ based on the following intuition. On the one hand, for different adversary strategies, the corresponding best response strategies involve different states and actions that will be added to $G(S', A')$ before the convergence of *IGRS*, and the number of iterations will be enormous if the strategy space is large. At each iteration, *LP* will be solved once to generate the adversary equilibrium strategy in $G(S', A')$. If we can sample some of these strategies, then we can reduce the number of iterations of *IGRS* and times to solve *LP*. On the other hand, if s is added to $G(S', A')$ (i.e., s is part of the best response), at least one of the paths in O_{h_s} is in the support set of the equilibrium strategy in $G(S', A')$. Then, it is possible that the adversary will escape through the paths in O_{h_s}, and the defender needs a corresponding strategy to interdict him. To generate such a strategy immediately, we can sample a uniform strategy over O_{h_s} and then call *BR*.

For using our technique to add multiple best response strategies at one iteration, the procedure of *IGRS* is shown in Figure 7.4. Specifically, before the convergence, we sample the uniform adversary strategies based on the states that are part of the best response and then call *BR*$(s_0, 0)$ at Line 4 in *IGRS* to compute the best response strategies against them. Here, S^* is the set of the states involved in the best response, A^* is the set of actions involved in the best response, and a uniform strategy over O_{h_s} means $y_o = 1/|O_{h_s}|$ $(\forall o \in O_{h_s})$. Especially, we sample a strategy with $y_o = 1/|O|$ $(\forall o \in O)$ at the initial step in *IGRS* (i.e., after s_0 is added to S') because we can compute the best response efficiently by using our effective pruning techniques including the mapping technique. Each strategy will be sampled at most once.

Finally, *IGRS* embedded with the above two additional techniques (*IGRS*++) is still optimal.

Theorem 7.7. IGRS++ *converges to an NE.*

Proof. This theorem is implied by Theorems 7.3–7.6. □

7.5 Experimental Evaluation

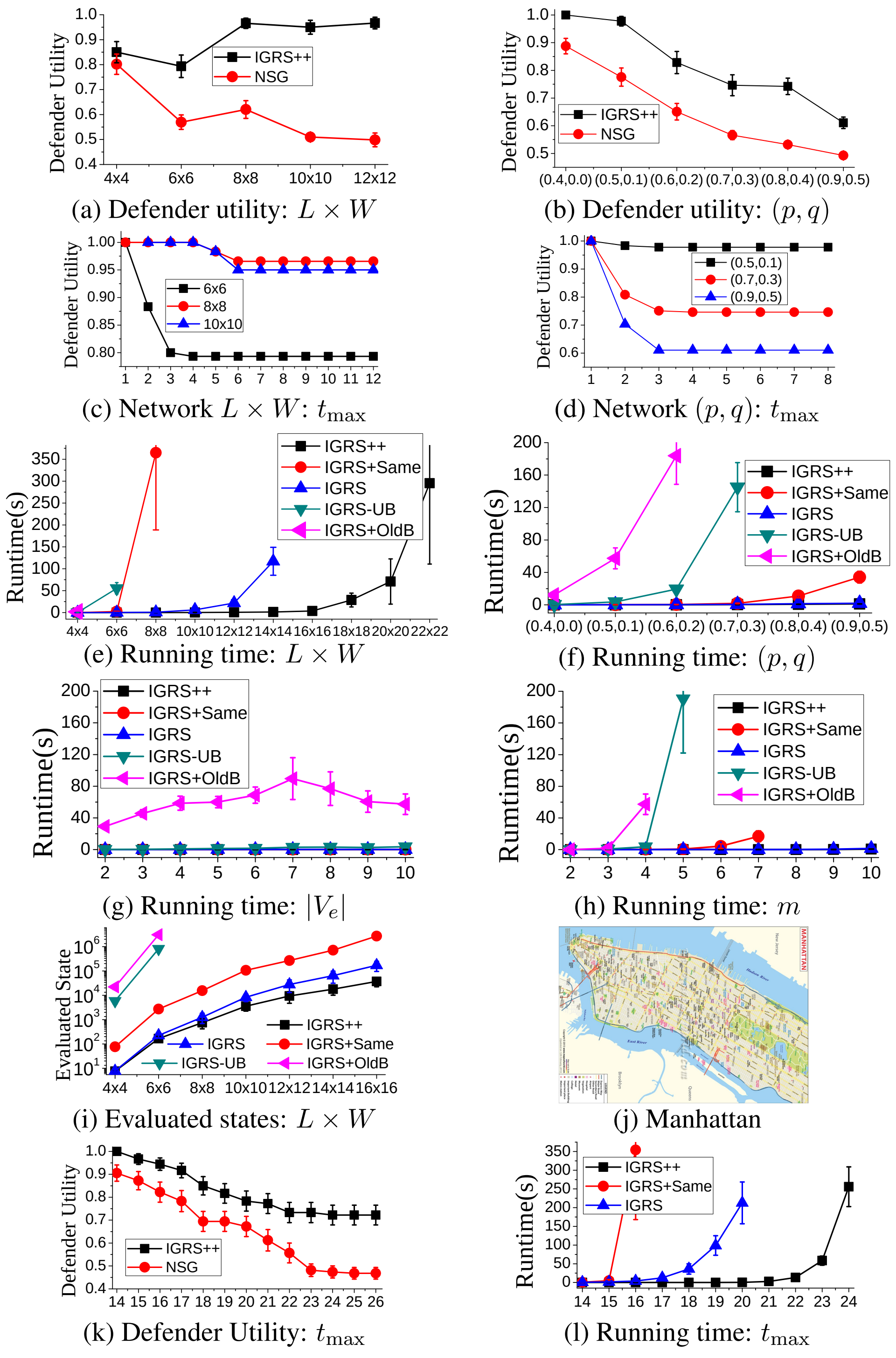
We demonstrate the efficiency of our algorithms through numerical experiments. We use CPLEX (version 12.6) to solve the linear program. All computations are performed on a machine with a 3.2GHz quad core CPU and 16GB memory. All results are averaged over 30 randomly generated instances. All random planar graphs are generated by the grid model with random edges [71], which samples an $L \times W$ square grid where horizontal/vertical edges between neighbors are generated with probability p, and diagonal ones with q. We choose p from $[0.4, 0.9]$ and q from $[0.0, 0.5]$ to match most real-world road networks. In our evaluation, v_0^a is located at the center node, m defender resources are uniformly distributed on the network at initialization, and $|V_e|$ exit nodes are randomly distributed at the border. By default, $L = W = 5$, $(p, q) = (0.5, 0.1)$, horizon $t_{\max} = L$, $|V_e| = 10$, and $m = 4$.

Solution Quality: We compare the solution quality of our approach (represented by *IGRS++*) with one baseline, the algorithm for the discussed network security game model [20, 22] considering time-dependent strategies, denoted as *NSG*. Figures 7.5a and 7.5b vary parameters: (1) the number of intersections, denoted by $L \times W$, and (2) edge generation probabilities. It can be seen that *IGRS++* significantly outperforms *NSG*. These results are similar to ones for varying the number of exit nodes or resources (not shown here). Here the defender's utility does not decrease with the size of the network because although the adversary has more strategies on larger networks, the defender can exploit more real-time information about him.

We evaluate the influence of the horizon $t_{\max}$ on the solution quality on networks with different scales in Figure 7.5c and networks with different density of edges in Figure 7.5d. Results show that the defender's expected utility converges when $t_{\max}$ is large enough, e.g., $t_{\max}$ is almost equal to L.

Scalability: We evaluate the scalability of *IGRS* with two baselines: (1) using the bounds setting in [38] for *IGRS* (*IGRS+OldB*), and (2) only using our new lower bound setting without the tight upper bound for *IGRS* (*IGRS-UB*). Specifically, *IGRS+OldB* can cut some branches only when the searching player's information set has at least two nodes. To be fair, each state s in *IGRS+OldB* is treated as a defender's information set, and each child history of h_s corresponds to one node in this information set. In addition, to evaluate our second technique at Section 7.4.3, we have a baseline, *IGRS+Same*, where *IGRS++* uses the method that adds several best response strategies against the same adversary strategy based on [26].

Results in Figures 7.5e–7.5h show that: 1) The runtime generally increases except that it reflects the "easy-hard-easy" pattern in security games [79] in Figure 7.5g. 2) *IGRS++* significantly outperforms *IGRS* and *IGRS+Same*, especially when the size of the problem is extremely large. 3) *IGRS* significantly outperforms *IGRS-UB*, and *IGRS-UB* significantly outperforms *IGRS+OldB*. To further evaluate the effect of our



(a) Defender utility: $L \times W$ (b) Defender utility: (p, q)

(c) Network $L \times W$: $t_{\max}$ (d) Network (p, q): $t_{\max}$

(e) Running time: $L \times W$ (f) Running time: (p, q)

(g) Running time: $|V_e|$ (h) Running time: m

(i) Evaluated states: $L \times W$ (j) Manhattan

(k) Defender Utility: $t_{\max}$ (l) Running time: $t_{\max}$

FIGURE 7.5: Solution quality: (a)–(b); Influence of $t_{\max}$: (c)–(d); Scalability: (e)–(i) (log y-axis in (i)); Real network: (j)–(l).

pruning techniques, Figure 7.5i shows the number of evaluated states for computing the best response against the adversary's uniform strategy, i.e., $y_o = 1/|O| (\forall o \in O)$, where results are consistent with the ones in Figures 7.5e–7.5h. Here, *IGRS+Same* runs slower than *IGRS* because its *BR* evaluates too many states to compute several best response strategies, and it adds too many strategies to the restricted game.

The Real-World Network: We also evaluate our approach on the road network of the whole Manhattan island of New York as shown in Figure 7.5j. In this network, there are 702 nodes with 1,216 links extracted from OSM (*www.openstreetmap.org*). The initial positions of four police officers and the adversary are chosen randomly, and seven exit nodes are chosen based on the connection to the external world of the island. A total of 30 cases with different adversary initial positions are tested. As shown in Figures 7.5k and 7.5l (note that *IGRS-UB* and *IGRS+OldB* cannot run on networks of this size), results are consistent with ones in Figures 7.5a–7.5i on random networks. In particular, *IGRS++* significantly outperforms others and efficiently solves NEST until the defender's expected utility converges.

7.6 Chapter Summary

This chapter studies the problem of optimal interdiction of urban criminals with the aid of real-time information. We show that ignoring such information can cause an arbitrarily large loss and then propose our NEST. We show that solving NEST is NP-hard, therefore develop an optimal algorithm *IGRS++* to solve it efficiently. Extensive experiments show the effectiveness of our approach in a wide range of settings.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

This thesis makes a thorough study of the TME computation in both normal-form and extensive-form zero-sum multiplayer games. Chapter 3–4 study this problem and its application in normal-form games, and Chapter 5–7 study this problem and its application in extensive-form games. In terms of algorithms, this thesis proposes the first incremental strategy generation algorithm guaranteeing to converge to a TME in Chapter 3. Then this thesis proposes various novel components for the incremental strategy generation algorithm to improve scalability: 1) the component on how to initialize the strategies in the restricted game in Chapter 3; 2) the component on how to efficiently solve the restricted game in Chapter 5; and 3) several best response oracles on how to efficiently compute the best response in various scenarios in Chapters 4, 6 and 7. The algorithms of this thesis are supported by theoretical and experimental results, followed by real-world applications where more specific restrictions or information is available. That is, the first contribution proposes general algorithms in normal-form games, while the second contribution is an application with domain-specific algorithms in normal-form games; and the third and fourth contributions propose general algorithms in extensive-form games, while the fifth contribution is an application with domain-specific algorithms in extensive-form games. Experiments show that our general algorithms are orders of magnitudes faster than baselines. In addition, our algorithms for applications are tested on real-world datasets, which shows that our algorithms can scale up to realistic problem sizes with hundreds of nodes on networks, including the real network of Manhattan.

The first contribution of this thesis is proposing an efficient ISG algorithm (CISGT) to converge to TMEs in normal-form games. We first study the properties of ISG for multiplayer games, showing that ISG converges to an NE but may not converge to a TME. Second, we design ISGT by exploiting that a TME is an NE maximizing the team's utility and show that ISGT converges to a TME and the impossibility of relaxing its conditions. Third, to further improve the scalability, we design CISGT by initializing the strategy space of ISGT via computing a CTME. Finally, experimental results show that CISGT is orders of magnitude faster than ISGT and the state-of-the-art algorithm compute TMEs in large games.

The second contribution of this thesis is proposing efficient algorithms to compute CTMEs in normal-form games for optimal escape interdiction on transportation networks. We show that finding a CTME is NP-hard. Therefore, we propose an efficient double oracle framework, *EIGS*, based on novel MILPs for best response oracles and heuristic algorithms for better response oracles. To solve large-scale problems, we develop *RedAD* to significantly reduce the strategy space for both players. Experimental results show that our algorithms obtain a robust solution that significantly outperforms baselines and can scale up to realistic-sized problems.

The third contribution of this thesis is proposing efficient algorithms to compute TMEs within any given accuracy in extensive-form games. We first show that the inefficiency of correlated strategies can be arbitrarily large in extensive-form games. To efficiently solve the non-convex program for finding TMEs directly, we develop novel global optimization techniques (i.e., an asynchronous precision method and an associated constraint method) and a novel iterative algorithm. Our algorithm is orders of magnitude faster than baselines in the experimental evaluation.

The fourth contribution of this thesis is proposing efficient algorithms to compute TMEsCor in extensive-form games. We first propose our novel HFS representation for the team, which does not change the set of equilibria in a game. We then develop our novel CMB including a finite upper bound for the number of iterations and a novel multilinear BR. In addition, we develop our novel global optimization technique ART to solve our multilinear BR exactly representing multilinear terms and efficiently generating associated constraints. Finally, experimental results show that CMB is orders of magnitude faster than baselines in large games.

The fifth contribution of this thesis is proposing efficient algorithms to compute TMEsCom in extensive-form games for optimal interdiction of urban criminals with the aid of real-time information. We show that ignoring such information can cause an arbitrarily large loss and then propose our novel game model NEST. We show that computing a TMECom is NP-hard in NEST, therefore develop an optimal algorithm *IGRS++* to solve it efficiently, which includes (i) novel pruning techniques in our best response oracle; and (ii) novel techniques for mapping strategies between subgames and adding multiple best response strategies at one iteration to solve extremely large problems. Extensive experiments show the effectiveness of our approach in a wide range of settings including the real-world network of Manhattan.

8.2 Future Work

In this section, we discuss several future directions.

The first direction is to develop efficient ISG algorithms to converge to TMEs in extensive-form games. Bosansky et al. (2014) show that we can exploit the feature of the game tree to develop efficient ISG algorithms to compute NEs by extending ISG in normal-form games. Inspired by this, based on our CISGT in normal-form games, we can explore to develop efficient ISG algorithms to converge to TMEs in extensive-form games by exploiting the feature of the game tree.

The second direction is to extend the counterfactual regret minimizing (CFR) algorithm [6] to compute TMEs. CFR is an important part for efficiently solving the Poker [88]. Unfortunately, it has been shown that CFR cannot guarantee to converge to an NE in multiplayer games [89], let alone a TME. However, inspired by our successful CISGT, we can explore to extend CFR to compute TMEs.

The third direction is to extend CFR to compute TMEsCom in our imperfect-recall game NEST with a combinatorial action space. CFR is built on the perfect-recall structure. However, we can dramatically reduce the number of states by using games with imperfect recall, similar to our NEST. On the other hand, CFR is based on the online learning algorithm [6], but the classic online learning algorithm does not consider the relation between actions, which will quickly reach the limit when the number of actions grows exponentially in some games. For example, in the escape interdiction game or in the pursuit-evasion game, where the defender/pursuer has a collection of teams, all of which can be deployed simultaneously, leading to a combinatorial action space. Consider just 10 teams with 5 actions (4 directions plus one action staying at the current node) for each to move at each intersection, where the potential number of combinatorial actions is $5^{10} \approx 10^7$, which is hard to update each action's counterfactual regret value in CFR.

The fourth direction is to extend the TME to capture more scenarios. For example, players in a team may not have the same utility function. In this case, the current formulation of computing TMEs may not work. Then we need to design the corresponding formulation and algorithm to compute the corresponding equilibrium. In extensive-form games, maybe, there are other forms of communication. For example, players in a team can have partial information during the play of the game (the communication level is between that in TMEsCor and that in TMEsCom). In this case, the current formulation of computing TMEs also may not work. Then we also need to design the corresponding formulation and algorithm to compute the corresponding equilibrium.

Bibliography

[1] John Nash. Non-cooperative games. *Annals of Mathematics*, pages 286–295, 1951.

[2] John Von Neumann and Oskar Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1953.

[3] Bernhard Von Stengel. Efficient computation of behavior strategies. *Games and Economic Behavior*, 14(2):220–246, 1996.

[4] Yoav Shoham and Kevin Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2008.

[5] H Brendan McMahan, Geoffrey J Gordon, and Avrim Blum. Planning in the presence of cost functions controlled by an adversary. In *ICML*, pages 536–543, 2003.

[6] Martin Zinkevich, Michael Johanson, Michael Bowling, and Carmelo Piccione. Regret minimization in games with incomplete information. In *NeurIPS*, pages 1729–1736, 2008.

[7] Vincent Conitzer and Tuomas Sandholm. Computing the optimal strategy to commit to. In *EC*, pages 82–90, 2006.

[8] Arunesh Sinha, Fei Fang, Bo An, Christopher Kiekintveld, and Milind Tambe. Stackelberg security games: Looking beyond a decade of success. In *IJCAI*, pages 5494–5501, 2018.

[9] Noam Brown and Tuomas Sandholm. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, 2018.

[10] Yang Cai and Constantinos Daskalakis. On minmax theorems for multiplayer games. In *SODA*, pages 217–234, 2011.

[11] Noam Brown and Tuomas Sandholm. Superhuman AI for multiplayer poker. *Science*, 365(6456):885–890, 2019. doi: 10.1126/science.aay2400.

[12] Xi Chen and Xiaotie Deng. 3-Nash is PPAD-complete. In *Electronic Colloquium on Computational Complexity*, volume 134, pages 2–29, 2005.

[13] Bernhard von Stengel and Daphne Koller. Team-maxmin equilibria. *Games and Economic Behavior*, 21(1-2):309–321, 1997.

[14] Nicola Basilico, Andrea Celli, Giuseppe De Nittis, and Nicola Gatti. Team-maxmin equilibrium: Efficiency bounds and algorithms. In *AAAI*, pages 356–362, 2017.

[15] Andrea Celli and Nicola Gatti. Computational results for extensive-form adversarial team games. In *AAAI*, pages 965–972, 2018.

[16] Youzhi Zhang and Bo An. Computing team-maxmin equilibria in zero-sum multiplayer extensive-form games. In *AAAI*, 2020.

[17] Albert Xin Jiang, Ariel D Procaccia, Yundi Qian, Nisarg Shah, and Milind Tambe. Defender (mis)coordination in security games. In *IJCAI*, pages 220–226, 2013.

[18] NYPD. Patrol. *https://www1.nyc.gov/site/nypd/bureaus/patrol/patrol-landing.page*, 2020.

[19] NYPD. Neighborhood policing. *https://www1.nyc.gov/site/nypd/bureaus/patrol/neighborhood-coordination-officers.page*, 2020.

[20] Manish Jain, Dmytro Korzhyk, Ondřej Vaněk, Vincent Conitzer, Michal Pěchouček, and Milind Tambe. A double oracle algorithm for zero-sum security games on graphs. In *AAMAS*, pages 327–334, 2011.

[21] Hiroaki Iwashita, Kotaro Ohori, Hirokazu Anai, and Atsushi Iwasaki. Simplifying urban network security games with cut-based graph contraction. In *AAMAS*, pages 205–213, 2016.

[22] Youzhi Zhang, Bo An, Long Tran-Thanh, Zhen Wang, Jiarui Gan, and Nicholas R. Jennings. Optimal escape interdiction on transportation networks. In *IJCAI*, pages 3936–3944, 2017.

[23] Youzhi Zhang, Qingyu Guo, Bo An, Long Tran-Thanh, and Nicholas R Jennings. Optimal interdiction of urban criminals with the aid of real-time information. In *AAAI*, volume 33, pages 1262–1269, 2019.

[24] Kristoffer Arnsfelt Hansen, Thomas Dueholm Hansen, Peter Bro Miltersen, and Troels Bjerre Sørensen. Approximability and parameterized complexity of minmax values. In *WINE*, pages 684–695. Springer, 2008.

[25] Manish Jain, Vincent Conitzer, and Milind Tambe. Security scheduling for real-world networks. In *AAMAS*, pages 215–222, 2013. ISBN 978-1-4503-1993-5.

[26] Zhen Wang, Yue Yin, and Bo An. Computing optimal monitoring strategy for detecting terrorist plots. In *AAAI*, pages 637–643, 2016.

[27] Aida Khajavirad and Nikolaos V Sahinidis. A hybrid LP/NLP paradigm for global optimization relaxations. *Mathematical Programming Computation*, 10(3):383–421, 2018.

[28] Gabriele Farina, Andrea Celli, Nicola Gatti, and Tuomas Sandholm. Ex ante coordination and collusion in zero-sum multi-player extensive-form games. In *NeurIPS*, pages 9638–9648, 2018.

[29] Andrea Celli, Alberto Marchesi, Tommaso Bianchi, and Nicola Gatti. Learning to correlate in multi-player general-sum sequential games. In *NeurIPS*, pages 13055–13065, 2019.

[30] NCVRW. *2016 NCVRW resource guide: Urban and rural crime fact sheet*. https://ovc.ncjrs.gov/ncvrw2016/content/section-6/PDF/2016NCVRW_6_UrbanRural-508.pdf, 2016.

[31] FBI. *Bank crime statistics 2016*. https://www.fbi.gov/file-repository/bank-crime-statistics-2016.pdf/view, 2017.

[32] Morgan Gaither, Mark Gabriele, Nancy Andersen, Sean Healy, and Vivian Hung. *Pursuit technology impact assessment, Version 1.1*. Final Report to the U.S. Department of Justice, 2017.

[33] Sara Marie McCarthy, Milind Tambe, Christopher Kiekintveld, Meredith L Gore, and Alex Killion. Preventing illegal logging: Simultaneous optimization of resource teams and tactics for security. In *AAAI*, pages 3880–3886, 2016.

[34] Xinrun Wang, Bo An, Martin Strobel, and Fookwai Kong. Catching Captain Jack: Efficient time and space dependent patrols to combat oil-siphoning in international waters. In *AAAI*, pages 208–215, 2018.

[35] Torrence D Parsons. Pursuit-evasion in a graph. In *Theory and Applications of Graphs*, pages 426–441. Springer, 1978.

[36] Yevgeniy Vorobeychik, Bo An, Milind Tambe, and Satinder P Singh. Computing solutions in infinite-horizon discounted adversarial patrolling games. In *ICAPS*, pages 314–322, 2014.

[37] Branislav Bošanskỳ, Albert Xin Jiang, Milind Tambe, and Christopher Kiekintveld. Combining compact representation and incremental generation in large games with sequential strategies. In *AAAI*, pages 812–818, 2015.

[38] Branislav Bosansky, Christopher Kiekintveld, Viliam Lisy, and Michal Pechoucek. An exact double-oracle algorithm for zero-sum extensive-form games with imperfect information. *Journal of Artificial Intelligence Research*, 51:829–866, 2014.

[39] Qingyu Guo, Bo An, Yair Zick, and Chunyan Miao. Optimal interdiction of illegal network flow. In *IJCAI*, pages 2507–2513, 2016.

[40] Scott Kolodziej, Pedro M Castro, and Ignacio E Grossmann. Global optimization of bilinear programs with a multiparametric disaggregation technique. *Journal of Global Optimization*, 57(4):1039–1063, 2013.

[41] Pedro M Castro. Normalized multiparametric disaggregation: An efficient relaxation for mixed-integer bilinear problems. *Journal of Global Optimization*, 64(4): 765–784, 2016.

[42] Tiago Andrade, Fabricio Oliveira, Silvio Hamacher, and Andrew Eberhard. Enhancing the normalized multiparametric disaggregation technique for mixed-integer quadratic programming. *Journal of Global Optimization*, 73(4):701–722, 2019.

[43] Xinrun Wang, Qingyu Guo, and Bo An. Stop nuclear smuggling through efficient container inspection. In *AAMAS*, pages 669–677, 2017.

[44] Jiří Čermák, Branislav Bošanskỳ, Karel Horák, Viliam Lisỳ, and Michal Pěchouček. Approximating maxmin strategies in imperfect recall games using A-loss recall property. *International Journal of Approximate Reasoning*, 93:290–326, 2018.

[45] Garth P McCormick. Computability of global solutions to factorable nonconvex programs: Part I–Convex underestimating problems. *Mathematical Programming*, 10(1):147–175, 1976.

[46] Yufei Wang, Zheyuan Ryan Shi, Lantao Yu, Yi Wu, Rohit Singh, Lucas Joppa, and Fei Fang. Deep reinforcement learning for green security games with real-time information. In *AAAI*, volume 33, pages 1401–1408, 2019.

[47] Marc Lanctot, Vinicius Zambaldi, Audrunas Gruslys, Angeliki Lazaridou, Karl Tuyls, Julien Pérolat, David Silver, and Thore Graepel. A unified game-theoretic approach to multiagent reinforcement learning. In *NeurIPS*, pages 4190–4203, 2017.

[48] Paul Muller, Shayegan Omidshafiei, Mark Rowland, Karl Tuyls, Julien Perolat, Siqi Liu, Daniel Hennes, Luke Marris, Marc Lanctot, Edward Hughes, et al. A generalized training approach for multiagent learning. In *ICLR*, 2020.

[49] Micah Adler, Harald Räcke, Naveen Sivadasan, Christian Sohler, and Berthold Vöcking. Randomized pursuit-evasion in graphs. In *ICALP*, pages 901–912, 2002.

[50] Richard Nowakowski and Peter Winkler. Vertex-to-vertex pursuit in a graph. *Discrete Mathematics*, 43(2):235 – 239, 1983.

[51] Milind Tambe. *Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned.* Cambridge University Press, 2011.

[52] Eric Shieh, Bo An, Rong Yang, Milind Tambe, Ben Maule, and Garrett Meyer. PROTECT: An application of computational game theory for the security of the ports of the United States. In *AAAI*, pages 2173–2179, 2012.

[53] Joshua Letchford and Vincent Conitzer. Solving security games on graphs via marginal probabilities. In *AAAI*, pages 591–597, 2013.

[54] Avrim Blum, Nika Haghtalab, and Ariel D Procaccia. Learning optimal commitment to overcome insecurity. In *NIPS*, pages 1826–1834, 2014.

[55] Yevgeniy Vorobeychik, Bo An, Milind Tambe, and Satinder P. Singh. Computing solutions in infinite-horizon discounted adversarial patrolling games. In *ICAPS*, pages 314–322, 2014.

[56] Yue Yin, Bo An, and Manish Jain. Game-theoretic resource allocation for protecting large public events. In *AAAI*, pages 826–834, 2014.

[57] Yue Yin, Haifeng Xu, Jiarui Gan, Bo An, and Albert Xin Jiang. Computing optimal mixed strategies for security games with dynamic payoffs. In *IJCAI*, pages 681–687, 2015.

[58] Mengchen Zhao, Bo An, and Christopher Kiekintveld. Optimizing personalized email filtering thresholds to mitigate sequential spear phishing attacks. In *AAAI*, pages 658–665, 2016.

[59] Nicola Basilico, Nicola Gatti, and Francesco Amigoni. Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In *AAMAS*, pages 57–64, 2009.

[60] Fei Fang, Thanh Hong Nguyen, Rob Pickles, Wai Y Lam, Gopalasamy R Clements, Bo An, Amandeep Singh, Milind Tambe, and Andrew Lemieux. Deploying PAWS: Field optimization of the protection assistant for wildlife security. In *IAAI*, pages 3966–3973, 2016.

[61] Jason Tsai, Zhengyu Yin, Jun-young Kwak, David Kempe, Christopher Kiekintveld, and Milind Tambe. Urban security: Game-theoretic resource allocation in networked physical domains. In *AAAI*, pages 881–886, 2010.

[62] Hong Seo Ryoo and Nikolaos V Sahinidis. Analysis of bounds for multilinear functions. *Journal of Global Optimization*, 19(4):403–424, 2001.

[63] Philipp C Wichardt. Existence of nash equilibria in finite extensive form games with imperfect recall: A counterexample. *Games and Economic Behavior*, 63(1): 366–369, 2008.

[64] Karel Horák and Branislav Bošanskỳ. A point-based approximate algorithm for one-sided partially observable pursuit-evasion games. In *GameSec*, pages 435–454, 2016.

[65] Karel Horák, Branislav Bosanskỳ, and Michal Pechoucek. Heuristic search value iteration for one-sided partially observable stochastic games. In *AAAI*, pages 558–564, 2017.

[66] Nicola Basilico, Nicola Gatti, and Francesco Amigoni. Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In *AAMAS*, pages 57–64, 2009.

[67] Nicola Basilico, Giuseppe De Nittis, and Nicola Gatti. Adversarial patrolling with spatially uncertain alarm signals. *Artificial Intelligence*, 246:220–257, 2017.

[68] Nicola Basilico, Andrea Celli, Giuseppe De Nittis, and Nicola Gatti. Coordinating multiple defensive resources in patrolling games with alarm systems. In *AAMAS*, pages 678–686, 2017.

[69] Albert Xin Jiang, Zhengyu Yin, Chao Zhang, Milind Tambe, and Sarit Kraus. Game-theoretic randomization for security patrolling with dynamic execution uncertainty. In *AAMAS*, pages 207–214, 2013.

[70] Alan Washburn and Kevin Wood. Two-person zero-sum games for network interdiction. *Operations Research*, 43(2):243–251, 1995.

[71] Wei Peng, Guohua Dong, Kun Yang, and Jinshu Su. A random road network model and its effects on topological characteristics of mobile delay-tolerant networks. *IEEE Transactions on Mobile Computing*, 13(12):2706–2718, 2014.

[72] Carlos F. Daganzo. The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory. *Transportation Research Part B: Methodological*, 28(4):269 – 287, 1994.

[73] Carlos F. Daganzo. The cell transmission model, part II: Network traffic. *Transportation Research Part B: Methodological*, 29(2):79 – 93, 1995.

[74] Alexander Skabardonis and Richard Dowling. Improved speed-flow relationships for planning applications. *Transportation Research Record: Journal of the Transportation Research Board*, 1572:18–23, 1997. doi: 10.3141/1572-03.

[75] S. Coogan and M. Arcak. A compartmental model for traffic networks and its dynamical behavior. *IEEE Transactions on Automatic Control*, 60(10):2698–2703, 2015.

[76] Jean-Patrick Lebacque. Intersection modeling, application to macroscopic network traffic flow models and traffic management. In *Traffic and Granular Flow'03*, pages 261–278. 2005.

[77] Konstantinos Aboudolas, Markos Papageorgiou, and Elias B. Kosmatopoulos. Store-and-forward based methods for the signal control problem in large-scale congested urban road networks. *Transportation Research Part C: Emerging Technologies*, 17(2):163 – 174, 2009.

[78] Traffic Assignment Manual. Bureau of public roads. *US Department of Commerce*, 1964.

[79] Manish Jain, Kevin Leyton-Brown, and Milind Tambe. The deployment-to-saturation ratio in security games. In *AAAI*, pages 1362–1370, 2012.

[80] Michael Johanson, Kevin Waugh, Michael Bowling, and Martin Zinkevich. Accelerating best response calculation in large extensive games. In *IJCAI*, pages 258–265, 2011.

[81] David R Morrison, Sheldon H Jacobson, Jason J Sauppe, and Edward C Sewell. Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning. *Discrete Optimization*, 19:79–102, 2016.

[82] Trevor A. Fischbach, Keo Hadsdy, and Amanda McCal. *Pursuit management: Fleeing vehicle tagging and tracking technology*. Final Report to the U.S. Department of Justice, 2015.

[83] Ashley Shook. Tracking "darts" take aim at police pursuits. *WWLP. http://wwlp.com/2017/03/22/tracking-darts-take-aim-at-police-pursuits/*, 2017.

[84] Paula Baker. Delta Police deploy new tool to help curb dangerous car chases. *Global News. http://globalnews.ca/news/3178224/delta-police-deploy-new-tool-to-help-curb-dangerous-car-chases/*, 2017.

[85] Brian A. Reaves. *Police vehicle pursuits, 2012-2013*. U.S. Department of Justice, Office of Justice Programs, 2017.

[86] Elizabeth Bondi, Fei Fang, Mark Hamilton, Debarun Kar, Donnabell Dmello, Jongmoo Choi, Robert Hannaford, Arvind Iyer, Lucas Joppa, Milind Tambe, et al. SPOT poachers in action: Augmenting conservation drones with automatic detection in near real time. In *IAAI*, 2018.

[87] David Reid. A real-time satellite-based surveillance of ships has gone live. *CNBC. https://www.cnbc.com/2018/07/19/leonardos-satellite-based-surveillance-of-ships-has-gone-live.html*, 2018.

[88] Noam Brown and Tuomas Sandholm. Safe and nested subgame solving for imperfect-information games. In *NIPS*, pages 689–699, 2017.

[89] Nicholas Abou Risk and Duane Szafron. Using counterfactual regret minimization to create competitive multiplayer poker agents. In *AAMAS*, pages 159–166, 2010.