Dynamic graph reduction optimization technique for interdiction games

ABSTRACT

Interdiction games, and related security games, have great potential in real world applications but face significant challenges in scaling up, both in the complexity of the player behavior as well as the size of the game. We develop a family of game reformulations that can be applied to a range of security games, and demonstrate their value in the context of an interdiction game. The reformulations exploit the relative sparsity of defensive assets across a large graph or physical space, leading to significant areas of low defender density. In our approach, a different reformulation is computed by the attacker at each iteration of a double-oracle solution to the game to maximize this effect. Our approach reduces solution time by two orders of magnitude, and allows solving complex interdiction games in a few hours using commodity hardware, where the attacker's strategy finds paths for multiple assets respecting resource constraints, on graphs with tens of thousands of nodes. We show empirically that the number of pure strategies in the defender's best strategy with a threshold probability mass grows slowly in the problem size, which is important to the success of our approach. Although developed for a specific interdiction game, the general approach is applicable to many security games with a relatively low defender density in the best mixed strategy during double oracle problem solving, and is complementary to other speed-up techniques, such as learned approximations to payoffs and MILP search control.

KEYWORDS

Game Theory, Game Optimization, Double Oracle, Interdiction Games

ACM Reference Format:

. 2022. Dynamic graph reduction optimization technique for interdiction games. In *Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), Auckland, New Zealand, May 9–13, 2022*, IFAAMAS, 7 pages.

1 INTRODUCTION

Interdiction games on graphs have many real-world applications, including in cybersecurity [11] and fighting smuggling [3]. Previously, solution techniques have been found that can scale to hundreds to thousands of nodes with a single attacker seeking a path through the graph avoiding a few blockade points, for example using a double oracle approach []. In order to apply solutions to interdiction games in more realistic scenarios, we aim solve games on grids with tens of thousands of nodes, and with tens of physically distinct attackers seeking paths avoiding tens of defenders. One approach might be to compute abstractions of the game space — smaller problems that can be solved much faster and whose solutions can be transformed efficiently into a solution to the original version of the problem. However, it is often infeasible to find an abstraction that will yield fast solutions against all the potential joint strategies of the attackers and defenders.

In this paper we present a *dynamic abstraction* approach, that exploits two observations about graph interdiction games with relatively large graphs compared with the number of attackers and defenders. First, given any one pure defender strategy that assigns an edge for each defensive asset, many of the edges in the graph will be undefended, and therefore many potential paths through the graph for attacker assets will have identical utility, both for the attacker and the defender. This fact alone does not lead to a useful abstraction of the graph, since when all potential defender strategies are considered, including probability distributions over pure strategies, a large set of links can be defended with some probability by at least one potential strategy. Our second observation, however, is that at any point during an iterative solution technique such as double oracle, each player typically tests a mixed strategy combining a relatively small number of pure strategies. Therefore, at each point during the solution process, the defender's current preferred strategy may leave portions of the graph where many alternative paths will have identical value in the game. These are not necessarily full paths for an attacker asset, but they could form part of an attacker path, and replacing this part with a single link simplifies the search problem for attackers and therefore reduces overall game solution time, which in graph interdiction games is typically dominated by the time required by the attacker oracle. Since the contribution to the utility for both attacker and defender is unaffected, the abstraction can be used to compute an exact solution to the original problem with no loss in solution quality.

Based on these observations, the dynamic abstraction approach computes a new abstraction of the problem at each iteration, that is specialized to the defender's current mixed strategy and that greatly reduces the cost for the attacker to generate a new best strategy. The approach is efficient when the total computational cost of computing a new abstraction, mapping the problem into the abstraction and mapping the solution back to a strategy in the original game space is still small compared with the cost of solving the original problem. We show empirically that this approach reduces solution time by two orders of magnitude in problems with thousands to tens of thousands of links, and shows much slower growth in the size of the problem indicating a reduction in overall complexity. The results are demonstrated in two domains: a family of physical graph interdiction games, and a family of cybersecurity games. However, we discuss conditions under which the approach may be expected to be successful in a broad range of interdiction games.

Key contributions of this paper include:

- A novel *dynamic abstraction* technique that uses graph reduction to significantly reduce the solution space of pure strategies in each iteration of the double oracle algorithm.

Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), P. Faliszewski, V. Mascardi, C. Pelachaud, M.E. Taylor (eds.), May 9–13, 2022, Auckland, New Zealand.

- The graph reduction technique preserves the solution quality and does not affect double oracle convergence to the defender expected utility corresponding to a Nash equilibrium of the original security game.
- We show empirically that the graph reduction technique reduces solution time by two orders of magnitude in physical graph interdiction games with thousands to tens of thousands of grid points, and by a factor of over 50 in a cybersecurity game with up to 10000 nodes.

In the next sections we discuss related work and describe the interdiction game that motivates our approach. Next we describe a dynamic abstraction algorithm for this domain and show experimental results. We conclude with a discussion of the potential breadth of this approach and future work to further improve and build on dynamic abstraction methods.

2 RELATED WORK

Jain et al. introduced a double oracle approach to scale the solution of zero-sum security games on graphs [4]. Haskell et al. [3] apply game theory for interdiction in open areas similar to those we consider and investigate alternate models of bounded rationality for the attackers.

Several researchers have developed approaches for continuous games, with continuous action spaces such as paths, rather than seeking discretizations of the problem that may be solved using mixed-integer linear programming. Kamra et al. [6] use neural networks to approximate players' best responses and expected payoffs. Their evaluation treats a discretization of the space as a proxy for ground truth, but this is computed with only one attacker and defender due to its complexity. The approach we describe here easily handles tens of attackers and defenders on a higher resolution grid in a similar domain. Lukas et al. [1] use a double oracle to compute Nash equilibria in continuous games, showing that it outperforms fictitious play. Other approaches to improve solution time include search control of the optimization process, *e.g.* [7]. These are complementary approaches to the game reformulations that we describe here, and could be applied in parallel.

Letchford et al. [9] explored security games on graphs and studied the effectiveness of computing the marginal probabilities of the defender resources in different settings (originally introduced by Kiekintveld et al. [8]). They showed that, while the approach is not always applicable, in some settings it gives a polynomial-time algorithm for computing an optimal defender strategy. This is also a complementary approach to the game reformulations that we describe in this paper.

Bsak et al. [2] introduced an algorithm that utilizes a subnet structure in a network to achieve a scalable double oracle game model. It uses an abstraction method for solving graph-based security games where all of the nodes except a restricted set of targets are removed, and then additional edges are added to preserve shortest paths. The targets are then added sequentially in a descending order of the target utility. This simplification may cover only a subset of targets. Our approach covers all targets.

3 INTERDICTION GAME AND DOUBLE ORACLE FRAMEWORK

We model interdiction games where two players each control multiple assets, moving across grids where motion in every direction is typically possible and the number of attackers and defenders is small compared to the size of the graph. The attacker divides a fixed amount of cargo between l vehicles, and seeks a set of paths from their initial locations to any of a set of m target locations. The value of a game for the attacker is the sum of cargo in vehicles that arrive at a target without being interdicted by the defender. The defender similarly controls a set of k vehicles and seeks a strategy represented as an assignment of each vehicle to a link. The value of a game for the defender is the value of cargo that is interdicted, as defined below.

An attacker strategy is a probability distribution of path assignments to each attacker vehicle. Interdiction takes place when a defender is assigned to one of the links on the attacker's path.

Our reformulations assume a solution style similar to the double oracle method [4]. This is an iterative approach where each player maintains its current best mixed strategy for the game. On each iteration, each player seeks a new pure strategy that is a best response to the opponent's current best strategy, typically through an optimization method such as mixed-integer linear programming. Once the new strategies are added, the best mix of strategies is computed and the next iteration begins. If a new best response cannot be found for either player, the solutions have reached an equilibrium.

The game is defined as follows:

- Graph $G = (N, E)$
- k defender resources
- l attacker resources $a_1 \dots a_l$
- m targets $t_1, \dots, t_m$
- payoff $\tau(j)$ (the amount of cargo assigned to attacker j)
- set of defender pure allocations $X = \{X_i\}, X_i = \{X_{ie}\} \forall e, X_{ie} \in \{0, 1\}$
- set of attacker paths $A = \{A_j\}, A_j = \{A_{je}\} \forall e, A_{je} \in \{0, 1\}$
- x = defender mixed strategy over X
- a = attacker mixed strategy over A
- $U_d(x, A_j)$ = defender expected utility playing x against A_j $U_d(x, A_j) = -\tau(j)\Sigma_i(1 - z_{ij})x_i$, where $z_{ij} = X_i \cap A_j$, $A_j = \{A_{je}\} \forall e, A_{je} \in \{0, 1\}$, $X_i = \{X_{ie}\} \forall e, X_{ie} \in \{0, 1\}$

We also model the range of an attacker asset, a distance covered for each edge traversal and a set of refueling points in the grid. If an attacker asset passes through a refueling point, its remaining range is reset to its maximum, modeling the pre-positioning of fuel at various locations by the attacker.

- range of attacker assets $r(a_i) > 0$
- distance on edges $d(e) > 0$ for $e \in E$
- refueling points: $f(n) \in \{0, 1\}$ for $n \in N$, where $f(n) = 1$ indicates a refueling point.

The defender seeks a placement of defender assets to maximize the expected interdicted payoff given the attacker's current best plan:

Defender oracle:

$$max_{z,\lambda} - \Sigma_{j,k}(1 - z_{jk})a_j\tau(a_{jk}) \quad (1)$$

subject to:

$$z_{jk} <= \Sigma_e A_{jke}\lambda_e, \qquad z_{jk} \in [0,1] \quad (2)$$

$$\Sigma_e \lambda_e <= k, \lambda_e \in \{0,1\} \quad (3)$$

Note: $z_{jk} \in [0,1]$ represents that the chosen assignment intersects the attacker path A_{jk}, where $A_{jk} = \{A_{jke} \forall e\}$. In final solution it will be set to either 0 or 1 since the objective function increases in z.

The attacker seeks a path for each asset that minimizes the expected interdicted value:

Attacker oracle:

$$max_{z,\gamma,\mathrm{lr}} \Sigma_{i,j}\tau(a_j)\Sigma_i x_i(1 - z_{ij}) \quad (4)$$

subject to:

flow constraints (enforcing a contiguous path for each attacker asset):

$$\Sigma_{e \in \mathrm{out}(n)}\gamma_e(a) = \Sigma_{e \in \mathrm{in}(n)}\gamma_e(a), \forall a, n \neq s, t^* \quad (5)$$

$$\Sigma_{e \in \mathrm{out}(s)}\gamma_e(a) = 1 \quad (6)$$

$$\Sigma_{j,e \in \mathrm{in}(t^*)}\gamma_e(a) = 1, \gamma_e(a) \in \{0,1\} \forall a \quad (7)$$

range constraints (enforces that no path exceeds the range of the asset without passing through a refueling point where $f(n) = 1$):

$$\mathrm{lr}(a,n) = r(a), \forall a, n \text{ where } f(n) = 1 \text{ or } n = s \quad (8)$$

$$\mathrm{lr}(a,n) <= \gamma_e(a)(\mathrm{lr}(a,n') - d(e)) + (1 - \gamma_e(a))r(a), \quad (9)$$

$$\forall n \text{ such that } f(n) = 0, e \in \mathrm{in}(n)(e = (n', n)) \quad (10)$$

$$\mathrm{lr}(a,n) >= 0 \forall a, n \quad (11)$$

interdiction constraints:

$$z_{ij} >= \gamma_e(a_j) + X_{ie} - 1, \forall i, j, e \quad (12)$$

$$z_{ij} >= 0, \forall i, j \quad (13)$$

Core LP:

$$max_{U_d^*, x} U_d^* \quad (14)$$

subject to:

$$U_d^* <= U_d(x, A_j) \forall j = 1, ..., |A| \quad (15)$$

$$1^T x = 1, x \in [0,1]^{|X|} \quad (16)$$

The Core LP finds an equilibrium of the restricted game between attacker and defender pure strategies (the definition of U_d is given above).

4 DYNAMIC ABSTRACTIONS FOR ITERATIVE GAME SOLVING

We seek to solve large games in order to provide strategies at meaningful resolutions for real-world problems. On the order of 25 time points are required to model a look-ahead at 100x100 resolution in a defender strategy, and around 50 at a preferable 200x200 resolution in domains of interest. Without problem reduction or reformulation, we would therefore seek to optimize sets of paths of length at least 25 through a graph with at least 10k nodes, which is not feasible. Previous solutions to similar problems have considered graphs of hundreds of nodes, typically with a significantly constrained topology, and have not considered multiple assets or paths for defenders. We therefore seek reformulations of the problem that reduce the number of alternative paths that must be examined using existing constraints and symmetries.

To motivate our approach to this problem, we make three observations. First, the computational requirements associated with finding a solution are dominated by the factorial explosion in the number of possible paths for each attacker as the grid size increases, as each path must be considered. Suppose an attacker path may go between points p and q in the grid. If p and q form a rectangle of side lengths a and b, there are $\binom{(a+b)}{a}$ possible paths between them, where

$$\binom{(a+b)}{a} = \frac{(a+b)!}{a!b!}.$$

However, given a defender mixed strategy x, if there exist paths between points p and q that have a zero probability of interdiction via x, then no other paths need be considered except the lowest cost (*i.e.* shortest) such path (unless refueling is required), in the sense that any solution for the attacker that incorporates points p and q will use one of these paths, and they are all equivalent in terms of expected payoff. A reformulation that treats all such paths interchangeably does not affect the value of the solutions found for either player, and still converges to a Nash equilibrium for the original problem. Below we refer to these paths as *dominant paths* with respect to the region defined by p and q we describe a region with dominant paths as a *dominated region.*

The second observation is that, although in principle a mixed strategy for the defender might include enough pure strategy elements that a defensive asset is defending every link with some probability, in practice the number of pure elements in the defender's best mixed strategy remains relatively low during the double oracle process, and therefore the number of links with non-zero defensive asset probability remains relatively low. For example, with 20 defenders in a grid on the order of 100x100, having on the order of 10k links, a mixed strategy consisting of 15 pure strategies can cover at most 3% of the links in the graph. Some areas of the graph may also remain without defensive assets due to constraints in the problem formulation, for example that some assets must remain within an area of jurisdiction, or cannot move far from an initial known position.

Therefore there are likely to be significant undefended regions in a defender's current best strategy at any point in the iterative double oracle process, perhaps with fewer such regions as the search approaches convergence. Our final observation is that approaches to identify and exploit these regions can be completed in time much smaller than the time to run the attacker oracle without using any graph reduction, and therefore it is feasible to use a different reformulation at each iteration of the double oracle approach, exploiting the undefended areas found in the defender's best strategy for that specific iteration. Below we show empirically that in typical runs, fewer than 20 pure strategies in the defender's best strategy contain 95% of the probability mass of the strategy, making our approach effective in this domain. (A strategy leaving large undefended portions of the graph can be the best plan for the defender, for example

concentrating its assets around key locations such as accessible targets or refueling points.)

These observations motivate a family of approaches for dynamic game abstraction in which at each iteration, parts of the search space are identified that can be shown to be equivalent in terms of the expected payoff against the adversary's current best strategy. These groups of equivalent choices are then replaced by a single choice when the oracle is initialized. If the player's new best strategy includes such a choice, one of the original solution pieces is substituted in the final solution (*e.g.,* a path in the original graph).

Our approach uses a heuristic to find effective abstractions of the game space for the attacker in the context of a defender's current best strategy. These abstractions correspond to dominated regions, within which all links are replaced by a single link corresponding to all dominant paths. The use of a heuristic helps ensure the reformulation is made relatively quickly on each iteration but does not affect solution quality, which is preserved by each dominant path reformulation. The tradeoff of the heuristic is that it may fail to find some dominated regions, leading to slower solution times than otherwise might be possible. We also note that the heuristic is specific to our game of interest, while the discussion up to here has been applicable more generally to a class of graph interdiction games. Similar heuristics exist for related games.

Our approach is as follows. First we create a reduced graph that contains only abstract links for dominated regions of three kinds: linking attacker source nodes to target nodes, linking attacker source nodes to refueling points and linking refueling points to target nodes. (In general, paths linking pairs of refueling nodes might be required but in problems of interest to us, at most one refueling step is needed.)

initialize a new, empty graph G'
initialize U as an empty set of un-reduced links
add all attacker initial locations, target locations and refueling points to G'
for all attacker initial locations a **do**
  for all refueling points or target locations p **do**
    if a and p define a dominated region **then**
      add edge(a, p) to G'
    else
      add edge(a, p) to U
    end if
  end for
end for
for all refueling points p **do**
  for all target locations l **do**
    if p and l define a dominated region **then**
      add edge(p, l) to G'
    else
      add edge(p, l) to U
    end if
  end for
end for

Some of the links considered do not represent dominated regions, meaning that there was no undefended path between the locations of interest. In this case we apply a simple heuristic search for a path that includes one defended link:

for all un-reduced links $(p, q) \in U$ **do**
  for all defended links d adjacent to p or q, ordered by probability in the mixed strategy **do**
    if a path exists consisting of undefended links $\cup\{d\}$ **then**
      add edge(p, q) to G' with cost reflecting the interdiction probability
    end if
  end for
  if no such path was found **then**
    add edge(p, q) to G' reflecting an arbitrary shortest path $p \to q$
  end if
end for

This approach can lead to suboptimal solutions, for example when the optimal path for an attacker asset carries a risk of interdiction away from either end point, or when the shared risk across multiple links is lower than that of other single links. In Section 6 we discuss tradeoffs between solution quality and speed in alternative approaches.

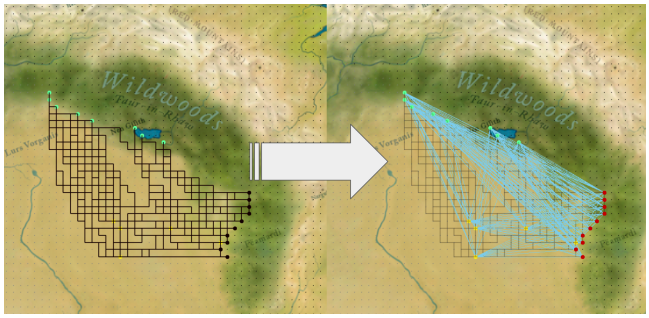

Figure 1: Graph reduction.

5 EXPERIMENTS

We conducted experiments in two settings: (1) a two-dimensional interdiction game with a grid-like graph and (2) an interdiction game for network and cybersecurity with a more densely connected graph. Figure 1 illustrates graph reduction approach for interdiction on the ground problem and Figure 7 illustrates three layers for the network for cybersecurity interdiction game.

5.1 Two-dimensional interdiction game

We created a game with 20 attackers, 5 defenders, 20 targets and 6 refueling points. We solved the game with and without using graph reduction on different grid resolutions. The game settings were translated into different grid resolutions. We used grid sizes from 50x50 to 150x150 with step size 5 (with total 21 data points). Games with different resolutions were created in such a way that the locations of the target nodes, attacker initial nodes and refueling points remain the same relative to the underling map regardless of the grid resolution. The payoff (amount of cargo assigned to each attacker asset) is uniformly distributed across all attacker assets.

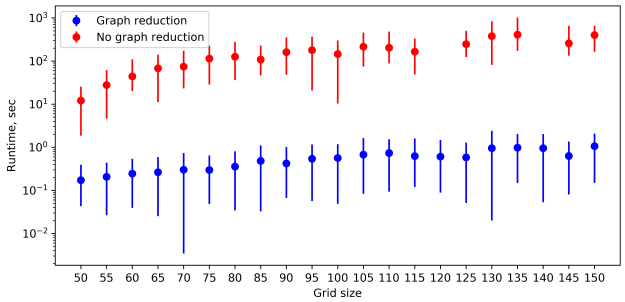

Figure 2: Average attacker oracle runtime per DO iteration.

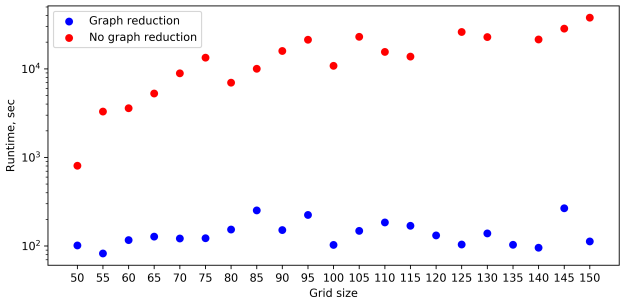

Figure 5: Overall game runtime till convergence

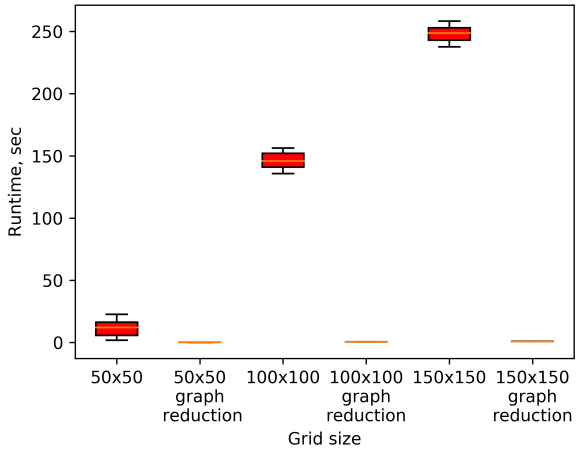

Figure 3: Attacker oracle runtime per DO iteration, linear scale

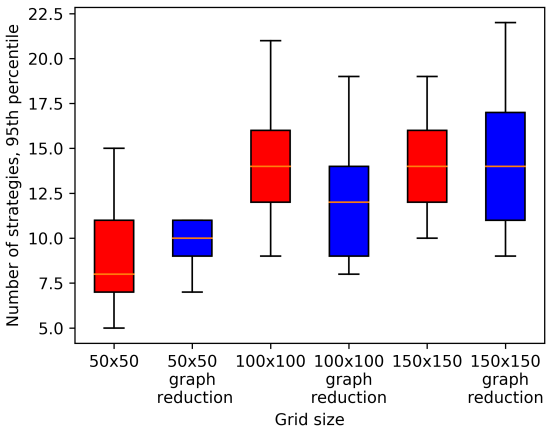

Figure 6: Number of pure strategies accounting for 95% of the probability mass in the best mixed strategy during iterative solving, computed by CoreLP

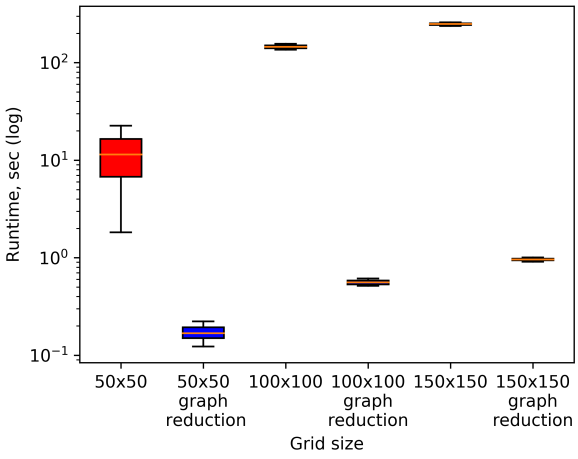

Figure 4: Attacker oracle runtime per DO iteration, log scale

Figure 2 compares average attacker oracle runtime per one iteration for games solved with and without graph reduction. In case of experiments without graph reduction two data points were excluded (grid sizes 120x120 and 135x135) because of the memory constraints and the large number of double oracle iterations (solution convergence was not reached before memory was exhausted). Figures 3 and 4 compare attacker oracle runtime per one iteration for 50x50, 100x100, 150x150 grids. The average attacker oracle runtime per one iteration decreases in games that utilize graph reduction by 200-300 times (see Figure 3 and 4).

Figure 5 shows the overall game runtime. Games with graph reduction run $\sim$ 100 times faster. Variance in the overall game runtime is caused by a different number of DO iterations needed to converge on the solution. Overall game runtime includes attacker, defender oracles runtimes, equilibrium solver (CoreLP) runtime and log keeping.

In addition to graph reduction we also used mixed strategy optimization where all pure strategies with near-zero probability are rounded to 0 (values in the 1-st percentile but not higher than 0.0001). This extends a set of edges that can be reduced by graph reduction. This is acceptable because the number of pure defender

strategies in the 95-th percentile with non-zero probability in pure mixed strategies found by CoreLP is usually between 5 and 20 (Figure 6).

The number of pure defender strategies that have non-zero weight and in 95-th percentile by weight (Figure 6) does not significantly change depending on grid size. It tends to be slightly lower for experiments with graph reduction.

We implemented DO framework with attacker and defender oracles according to formulation in section 3. To solve optimization problems we used Gurobi solver [10].

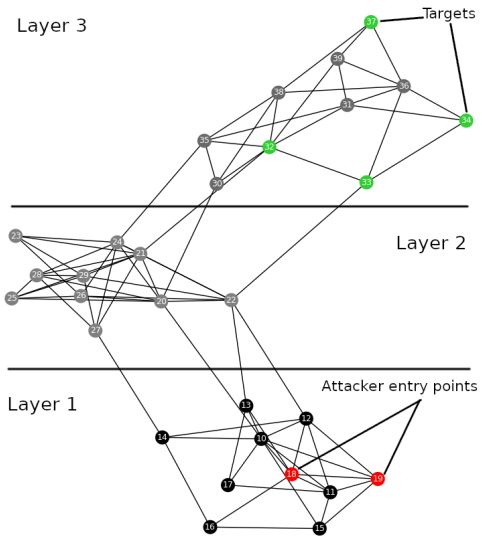

Figure 7: Three layers for the network for cybersecurity interdiction game. Green nodes represent targets, red nodes represent attacker entry points, black and grey nodes mark nodes of different network layers

5.2 Cybersecurity interdiction game

In a cybersecurity interdiction game the computer network is a graph where every node represents a machine that provides one or more services. Edges in the graph represent links between services. For example, access control between services, physical network connections between machines, etc. Targets are nodes of interest for attackers. Defenders allocate defense tools (firewalls) on links between nodes to interdict attacks. Topologically these graphs usually are represented as connected hierarchical layers (see illustration in 7).

We created a game with 20 entry points for attackers (20 attackers), 5 defenders (only 5 edges can have defense tools working at a time), 20 target nodes and 6 access control points (similar to refueling points in the interdiction on the ground game setup). The network has three layers. Each layer is a highly interconnected structure where the probability of an edge between any two nodes in one layer is 0.7. Cuts between layers have 1/10 of the number of nodes in the layer connected to another layer. We solved the game with and without using graph reduction on different grid resolutions. The game settings were translated into different graph sizes. We used graphs with 250, 500, 1000, 2000, 4000, 10000 nodes. The payoff for reaching each of the targets is uniformly distributed across all attackers.

Figure 8 compares average attacker oracle runtime per one iteration for games solved with and without graph reduction. Figure 9 shows the overall game runtime. Games with graph reduction are solved on average ~ 52 times faster. Variance in the overall game runtime is caused by a different number of DO iterations needed to converge on the solution. Overall game runtime includes attacker, defender oracles runtimes, equilibrium solver (CoreLP) runtime and log keeping.

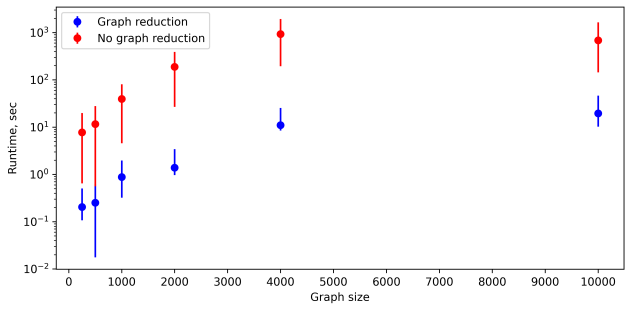

Figure 8: Attacker oracle runtime per one DO iteration, log scale

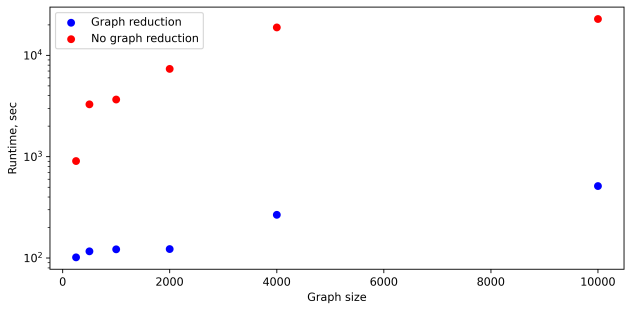

Figure 9: Overall game runtime till convergence in cybersecurity interdiction game

Attacker and defender oracles were implemented according to the formulation in section 3.

6 SUMMARY, DISCUSSION AND FUTURE WORK

We have demonstrated a dynamic game reformulation approach that produces a speed-up of over 50-100x on real-world problems. Our approach was applied to solving problems in two domains: the two-dimensional interdiction game and the network cybersecurity game. The interdiction game in both domains was solved with multiple attacker and defender assets, complex path constraints on attackers including planning to refuel (interdiction game the ground) and to gain access control nodes (network cybersecurity), and graph sizes in the tens of thousands of nodes and edges in a few hours on regular hardware. In addition, the runtime grows more slowly in the problem size compared with the unmodified problem. The approach can be used concurrently with other speed-up approaches such as learned rules for MILP solvers [7]. A novel aspect of the approach is its exploitation of the current best mixed strategy of the adversary, essentially computing a different abstraction at each iteration of the double oracle process.

Although component reformulations are designed to preserve solution quality (double oracle convergence to the defender expected utility corresponding to a Nash equilibrium of the game), the technique we describe here computes an approximate solution in each iteration due to its heuristic approach when undefended paths cannot be found. The heuristic solution underestimates the value of the potential best attacker strategy. An alternative approach might yield an exact solution, at the expense of greatly reduced speed-up in problem solving, by combining sections of the graph at the original granularity where defenders are concentrated with abstracted links where they are absent. We plan to explore the tradeoffs in runtime and accuracy within the family of dynamic game reformulation methods by developing heuristic approaches that exploit the structure of the graph regions of interest, for example reasoning about the cutset of defended links when no path exists, or dropping low-probability pure strategies from the mixed defensive strategy as a coherent simplification, rather than reasoning about the combined probability mass of individual links.

Although we develop and test our approach in the context of two specific interdiction games, we believe the general approach described here is applicable to a wide range of green security games that are likely to have significant undefended portions of the graph in the defender's intermediate strategies. This includes many interdiction games with geographic constraints, such as the forest protection game [5] and others. The key elements are a limited number of defensive assets in each pure defensive strategy, relative to the size of the graph, and a mixed strategy that is concentrated in a relatively small number of pure strategies. Although the attacker's goal may vary from reaching one of a set of target destinations and the geographical distribution of payoff is quite different, we believe adaptations of this approach to those domains may lead to significant improvements in runtime and feasible problem size.

REFERENCES

[1] Lukáš Adam, Rostislav Horčík, Tomáš Kasl, and Tomáš Kroupa. 2021. Double oracle algorithm for computing equilibria in continuous games. *Proc. of AAAI-21* (2021).

[2] Anjon Basak, Fei Fang, Thanh Hong Nguyen, and Christopher Kiekintveld. 2016. Abstraction Methods for Solving Graph-Based Security Games. In *Autonomous Agents and Multiagent Systems*, Nardine Osman and Carles Sierra (Eds.). Springer International Publishing, Cham, 13–33.

[3] William Haskell, Debarun Kar, Fei Fang, Milind Tambe, Sam Cheung, and Elizabeth Denicola. 2014. Robust protection of fisheries with compass. In *Twenty-Sixth IAAI Conference*.

[4] Manish Jain, Dmytro Korzhyk, Ondřej Vaněk, Vincent Conitzer, Michal Pěchouček, and Milind Tambe. 2011. A double oracle algorithm for zero-sum security games on graphs. In *The 10th International Conference on Autonomous Agents and Multiagent Systems*. 327–334.

[5] Nitin Kamra, Umang Gupta, Fei Fang, Yan Liu, and Milind Tambe. 2018. Policy learning for continuous space security games using neural networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[6] Nitin Kamra, Umang Gupta, Kai Wang, Fei Fang, Yan Liu, and Milind Tambe. 2019. DeepFP for Finding Nash Equilibrium in Continuous Action Spaces. In *Decision and Game Theory for Security*, Tansu Alpcan, Yevgeniy Vorobeychik, John S. Baras, and György Dán (Eds.). Springer International Publishing, Cham, 238–258.

[7] Elias Khalil, Pierre Le Bodic, Le Song, George Nemhauser, and Bistra Dilkina. 2016. Learning to Branch in Mixed Integer Programming. *Proceedings of the AAAI Conference on Artificial Intelligence* 30, 1 (Feb. 2016). https://ojs.aaai.org/index.php/AAAI/article/view/10080

[8] Christopher Kiekintveld, Manish Jain, Jason Tsai, James Pita, Fernando Ordónez, and Milind Tambe. 2009. Computing optimal randomized resource allocations for massive security games. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. 689–696.

[9] Joshua Letchford and Vincent Conitzer. 2013. Solving security games on graphs via marginal probabilities. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*.

[10] Bernhard Meindl and Matthias Templ. 2012. Analysis of commercial and free and open source solvers for linear optimization problems. *Eurostat and Statistics Netherlands within the project ESSnet on common tools and harmonised methodology for SDC in the ESS* 20 (2012).

[11] Apurba K Nandi, Hugh R Medal, and Satish Vadlamani. 2016. Interdicting attack graphs to protect organizations from cyber attacks: A bi-level defender–attacker model. *Computers & Operations Research* 75 (2016), 118–131.