1. This is about Patrolling Security Games

(alarm system)

The game any of the two following events. The first one is when D patrols a target t that is under attack by A from less than d(t) turns. In such case the attack is prevented and A is captured.

The second one is when target t is attacked and D does not patrol t during the d(t) turns that follow the beginning of the attack. In such case the attack is successful and A escapes without being captured.

When A is captured, D receives a utility of 1 and A receives a utility of 0. When an attack to t is successful, D receives 1- pi(t) and A receives pi(t).

The game may not conclude if A waits for every observed position of D, and never attacks. In such case, D receives 1 and A receives 0.

The game is constant sum and therefore it is equivalent to a zero-sum game through a positive affine transformation.

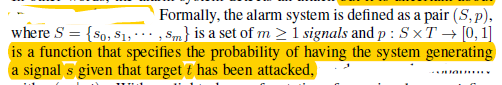
Here, D is the leader and A is the follower. Since we focus on zero-sum

games, the leader’s strategy at the leader-follower equilibrium is its maxmin strategy and it can be found by employing linear mathematical programming, which requires polynomial time in the number of actions available to the players.

1. It discussed uncertainty in security game model
   1. (alarm system with spatial uncertainty)

The alarm system uses a number of sensors spread over the environment to gather information about possible attacks and raises an alarm signal at any time an attack occurs.

The alarm system detects an attack but it is uncertain about the target under attack.



Two examples of alarm systems for patrolling setting

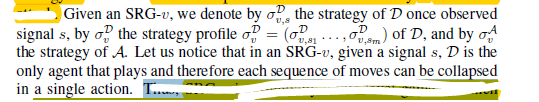
1. Low-accuracy alarm system that generates the same signal anytime a target is under attack, not spatial information
2. A more accurate info about the location of the attack
   1. when ti is attacked the alarm system generates si with high probability

The game tree and it’s composition (page 64)

When alarms goes off, D finds the best strategy starting from vertex v to responds that signal, using signal response game. SRG-v.

* + 1. Signal response game

SRG-v is essentially a two-level game in which A decides the target to attack and D decides the sequence of moves on the graph.



For

After that, design algorithms for the equilibrium of the patrolling game.

Need to compliance 4 questions

“Question 1. Which is the best patrolling strategy for D maximizing its expected utility?

Clearly, this problem is related to what we called PG in our game decomposition.

In order to build an answer, we pose other three questions that, instead, involve the other subgame called SRG-v.

Question 2. Given a starting vertex v and a signal s, is there any strategy allowing D to visit all the targets in T(s), each within its deadline?

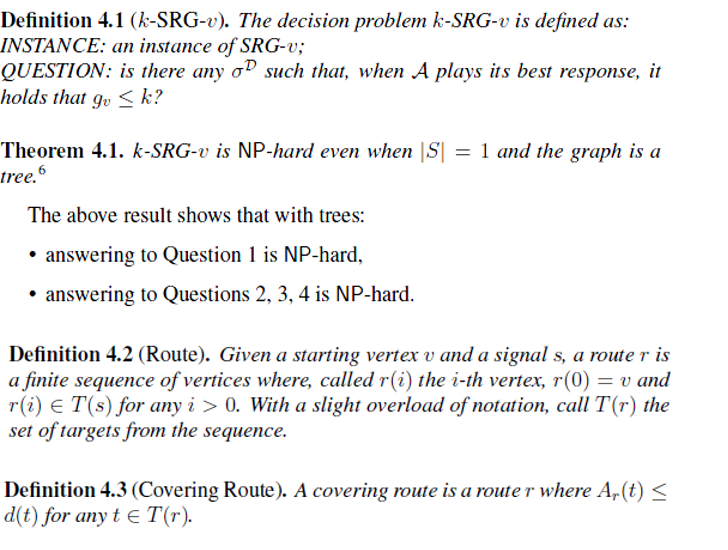
Question 3. Given a starting vertex v and a signal s, is there any pure strategy giving D an expected utility of at least k?

Question 4. Given a starting vertex v and a signal s, is there any mixed strategy giving D an expected utility of at least k?

“

Discussed the computational complexity of finding an exact or approximate equilibrium of the game model

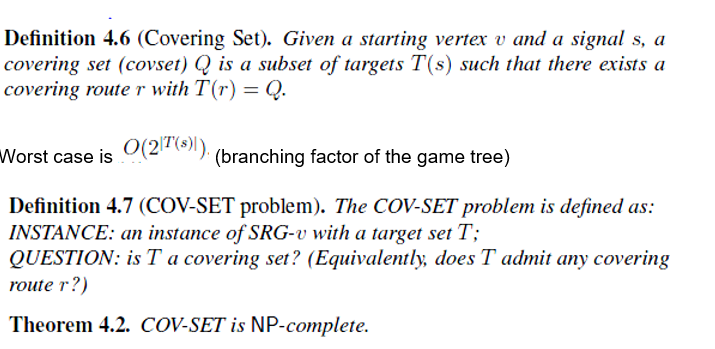
To find the equilibrium, need to find the best patrolling strategy, here it’s finding the set of best routes (covering route). Find covering set.



Proof of theorem 1 in appendix

For covering routes, two definitions, intra set dominance and Inter-set dominance

Cuz of intra set dominance, can use only one route per covering set.



According to inter set dominance, Find the maximal covering set, MAX\_COV\_SET (exponential time)

Can compute covering sets with

1. Dynamic Programming
2. Branch and bound

For Dynamic Programming , DP-ComputeCovSets (exact algo), to compute the stretagies available to D, but this cannot approximate maximal covering route.

Approximate algorithm – MonotonicLongestRoute to find the maximal covering route.

For Branch And bound-

Exact Algorithm

When penetration time are relax and good heuristics will make depth first search find the maximal covering route firster than breath-first search.

The algorithm is a tree search using two global variables mainly CLmin and CLmax

The covering routes will be returned in CLmax while CLmin is used for pruning.

Algorithm 8, Tree-search is used for traversing the tree. If it is an open branch, it continues to call Tree-search recursively until the r is fully expanded.

In each open branch, when the depth of node in the tree is smaller or equal to p{T(s)} then backtracking is disabled.

with \_ = 0 we do not rely on the heuristics at all, full backtracking is enabled, the tree is fully expanded and the returned R is complete, i.e., it contains all the non-dominated covering routes. Route r is repeatedly expanded in a greedy fashion until no insertion is possible.

Pruning is done with Close(). Where CLmin minimal set of close routes is used.

General idea: a closed route r belongs to CLmin only if CLmin does not already contain another r0 \_ r.

Once a route r is closed, 4’ is inserted to CL min without checking. This is safe cuz of the definition

CL max maintains a set of the generated maximal closed routes.

Final soltion

The thesis presented Expand algorithm for route expansion which partitioned the targets T(s) into Ttight and Tlarge where d(t) < delta\* w\*v,t is Ttight and Tlarge otherwise.

The insertion of target for expanded nodes works with the following rules:

* insertion of a target belonging to Ttight is always preferred to the insertion of a target belonging to Tlarge, independently of the insertion position;
* insertions of t 2 Ttight are ranked according to h considering first the insertion position and then the target;
* insertions of t 2 Tlarge are ranked according to h considering first the target and then the insertion position.

This has exponential computational complexity since it builtis full tree of covering routes

Approximate Algorithm

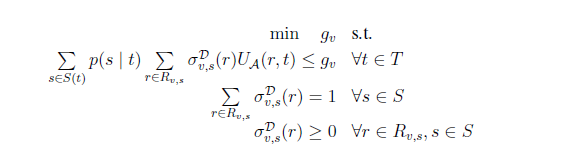
P denote completeness and p<1 in favour of a less computational effort

When p is k/|T(s)|, the complexity becomes polynomial in size of input theorem 4.9

Solving srg-v

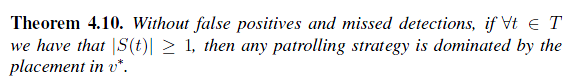
Will compute the optimal signal response strategy for D.

According to the linear program below with U(r,t) = pi(t) if t not in r and 0 otherwise,



Hardness is polynomial since it has |T|+|S| constraints and O(|V||S|maxv,s|Rv,s|) variables. And it only depends on max (term) and this only depends on |T(s)|

Patrolling Game:



When no false positives and no missed detections are present, the optimal Defender strategy is to stay in a fixed location, wait for a signal, and respond to it at best. This strategy keeps being optimal even when non-negligible missed detection rates are allowed.

So, Theorem 4.10 simplifies the computation of the patrolling strategy by reducing it to the problem of finding v\*. To such aim, we must solve a SRG-v for each possible starting vertex v and select the one with the maximum expected utility for D. Algorithm 11 depicts the solving The Function SolveSRG(v) returns the optimal value 1 – gv\* .

The simulations for real life case (Expo) show that the optimal patrolling strategy coincides with such fixed placement even under false negatives rates of at least approximately equals to 0.3.

1. Miss detections
   1. Problem formulation

p(s | t) = the probability of generating a signal s given that target t has been attacked s0 = null signal corresponding to the absence of alarms

si (i > 0) = an alarm signal caused by some attack

Thus p(s0 | t) = alpha for every target t.

The game being constant sum, the best leader’s strategy is its maxmin/minmax strategy.

There are two defender strategy (sigma p D and signma a D)

Sigma p D is patrolling strategy is adopted when no alarm signal is received, s0 is received.

Sigma a D is signal response strategy when alarm signal is received.

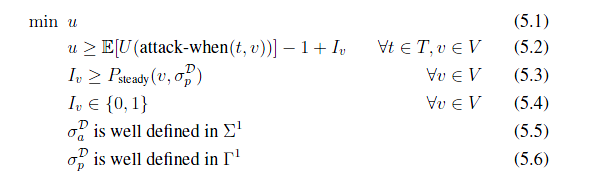
* 1. special type of patrolling strategy, covering cycle

Where alpha = 0, the optimal patrolling strategy is indeed a placement. However, this no longer holds with a non-null missed detection rate, even with very small values.

Here,

If alpha > 0, any placement-based strategy will fail to capture attacks in the occurrence of a false negative.

Finding minmax strategy can be formulate with a non-linear mathematical programming formulation with l=1:

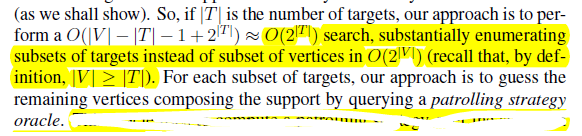


(what is Iv here?? Why it is non-linear non convex? Page 109)

Unlike, the algorithms without missing detection, Problem (5.1)–(5.6) cannot be separated into two independent programs, one searching for the best \_Dp and another for the best \_Da : in fact, doing so would inevitably provide sub-optimal patrolling and signal response strategies.

1. Resolution Approach

Support graph of optimal patrolling strategy



* + - 1. Patrolling strategy oracles
         1. Covering Cycle oracle
         2. Random Oracle
      2. Signal response oracle
      3. Target selection heuristics
         1. 1 dynamic approach (greedy)
         2. 2 static approach (on target values) and on target distance

Covering cycle oracle

This oracle, tries to compute a covering cycle protecting a given subset of targets T0. If found, no better patrolling strategy can be given for any support containing the same targets since it always guarantees capture on them, eve in presence of missed detections.

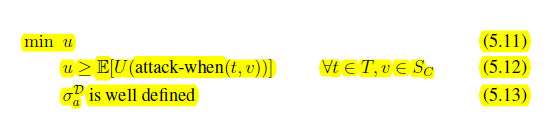
The CCO must solve a PSPACE-hard problem, so no exact efficient method can be designed. We resort to the approach where an algorithm for such problem is provided with correctness guarantees. The algorithm limits the solution length to kjV j (k is a parameter) and applies a backtracking search. The algorithm is still exponential in the worst case (even for a fixed k), but empirically it demonstrated acceptable performance on realistic settings, even with large numbers of targets.

Random Oracle

The Random oracle (RO) computes the optimal patrolling strategy. This problem is the equivalent to the computation of the optimal randomized patrolling strategy as done in [22] and can be done with non-linear programming techniques.

The RO is meant to work as a backup and is run in parallel to the CCO (see Figure 5.3). When the CCO terminates with no answer or reaches a timeout, we check if RO was able to terminate and, in the positive case, we fetch the patrolling strategy it computed and continue with our search.

Imp



Implemented by two

• DP-ComputeCovSets (exact)

• MonotonicLongestRoute (approximate)

Both based on dynamic programming

Target Selection Hueristics

The first one follows a dynamic approach in the sense that the ranking of possible solutions to explore changes as new subsets are explored.

Dynamic Method (greedy search):

if we searched subset T0 and we obtained strategy \_0, then constructing a

T00, obtained by including in T0 targets that provide A with large expected utilities under \_0 and searching it, might result in a better strategy \_00.

With this method, we progressively include in the support targets that are the most strategically appealing to the Attacker.

Two static approaches where the order of preference between the candidate solutions is fully determined by the problem instance and does not vary during the search process. By ranking targets according to their value or by ranking targets according to target distance.

1. Multiple resources for defense and attacks

Our security game allows the Defender to control an arbitrary number of resources, denoted by m, instead of just a single one.

* 1. Minimizing number of defensive resources
     1. Computing the minimum covering placement is in log-APX.
     2. But for tree and cycle graphs – it is solvable by polynomial time
  2. Covering routes are computed (same as single resource)
  3. Signal Response strategy
     1. 3 signal response oracles
        1. Fully coordinated SRO

The game can be solved computing the maxmin strategy by linear programming.

* + - 1. Partially Coordinated SRO

Non-Linear Program (NLP) whose size is linear in the number of resources compressing exponentially the size of the game

* + - 1. No coordinated SRO

(NC-SRO is not Coordination at all it’s just multiple instances of single SRO agents)

(FC-SRO > PC-SRO)

1. Multiple attacks
   1. Simulatenous attacks
   2. Sequenctial attacks
      1. Simultaneous attacks < sequential attacks for attacker gain

Resolution approach with coordinated defense

1. Robustness to the wrong guess
   * 1. Overestimating and underestimating (both has loss for defender)
     2. Competitive factor (worst case ratio v/v\*) for defender

Defender gets competitive factor of 0 for the wrong guess (both underestimating and overestimating).

1. Deploy online algorithms???

For deterministic, the best competitive factor is 1/k-1 where k is attacker resources

Sometimes randomization will give better competitive factor than deterministic algorithms.

Learning attacker’s profile

* 1. Analysed attacker profiles
     1. Stochastic attacker
     2. Strategy aware attacker
     3. SUQR Attacker

If the defense system is targeted for stochastic attacker and the attacker became stackelberg attacker there is a loss for defender (vice versa)

For the loss, loss of stretagy aware algo > stochastic algo

* 1. Identifying the attacker
     1. Follow belief
     2. Follow regret

Follow belief is more efficient

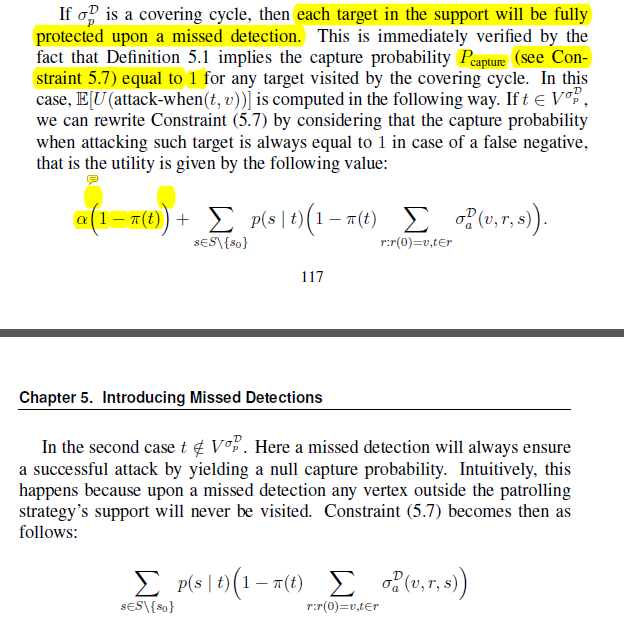
1. MAB, FPL, UCB

Comments:

It would be better if the thesis included a definition for PSPACE and PSPACE hard too.

Regarding computational complexity theory, I want to learn more about it.

I cannot understand this



Page 156, Algorithm 14 , line 4 (shouldn’t it be min R and min U)?

I

I have a doubt regarding missing detection rate, in Chapter 4 it is said that

“The simulations for real life case (Expo) show that the optimal patrolling strategy coincides with such fixed placement even under false negatives rates of at least approximately equals to 0.3.” –quote, page 101 Proof (Proposition 4.11)

Yet in chapter 5, “If missing detection rate is greater than 0, any placement-based strategy will fail to capture attacks in the occurrence of a false negative.” Quote- page 108 Proof (Proposition 5.2)

Again with theorem 5.5’s proof “Every patrolling strategy inducing a support with less than two targets is weakly dominated by a static placement strategy, that is a patrolling strategy with a singleton support.” Page 112-113 , it has proof that for 1 target, A will always be captured.

So I think page 108 should mention the number of targets.

Future ??

The best placement strategy for other games???