Exploiting Adversary's Risk Profiles in Imperfect Information Security Games

Gabriel Stocco and George Cybenko

THAYER SCHOOL OF ENGINEERING AT DARTMOUTH

GameSec
November 14, 2011

Table of Contents

- In my thesis studies and in our group we are studying techniques for learning an adversary's behavior.
- Today I'll be discussing one aspect of that broader problem which I have been investigating.

Problem Statement

- *What is an imperfect information game?*

Problem Statement

- *What is an imperfect information game?*
- A game in which players do not have the same information that they would have at the end of the game. For example, any card game.

Problem Statement

- *What is an imperfect information game?*
- A game in which players do not have the same information that they would have at the end of the game. For example, any card game.
- *Why are such games interesting?*

Problem Statement

- *What is an imperfect information game?*
- A game in which players do not have the same information that they would have at the end of the game. For example, any card game.
- *Why are such games interesting?*
- There are real world security scenarios of interest, which may closely resemble - and thus be good candidate for modeling by - an imperfect information game.

Why are such games interesting?

- For example, in international diplomacy, between multiple nations. The nations each have secret information they have discovered about the other nations. When conflict occurs, they may choose to reveal information, embarrassing the other players and causing loss of credibility (and gain for themselves).

Why are such games interesting?

- For example, in international diplomacy, between multiple nations. The nations each have secret information they have discovered about the other nations. When conflict occurs, they may choose to reveal information, embarrassing the other players and causing loss of credibility (and gain for themselves).
- In computer security, where you may need to change your defense, or offense strategy based on the skill, or appetite for risk, of your adversary.

Previous Work

- Traditional normal form game analysis often requires a priori knowledge of game payoffs, and can produce trivial Nash equilibrium results.

Previous Work

- Traditional normal form game analysis often requires a priori knowledge of game payoffs, and can produce trivial Nash equilibrium results.
- For example in Lye et al. 2005 the authors propose a game with a small finite action space for both the attacker and defender, and a priori assumptions about the skill of the attacker and payoffs.

Previous Work

- Traditional normal form game analysis often requires a priori knowledge of game payoffs, and can produce trivial Nash equilibrium results.
- For example in Lye et al. 2005 the authors propose a game with a small finite action space for both the attacker and defender, and a priori assumptions about the skill of the attacker and payoffs.
- In this case, one of the assumptions is that the attacker has a 90% chance to compromise a workstation, and doing so is worth 50 utility.

Previous Work - Lye

You can then obtain equilibrium strategies for each player, as well as the payoffs resulting from that equilibrium. However, this equilibrium is only meaningful in the context of the specific a priori values assumed at the very beginning of the analysis!

		Strategies		State Values	
	State	Attacker	Administrator	Attacker	Administrator
1	**Normal_operation**	[1.00 0.00 0.00]	[0.33 0.33 0.33]	210.2	-206.8
2	**Httpd_attacked**	[1.00 0.00 0.00]	[0.33 0.33 0.33]	202.2	-191.1
3	**Ftpd_attacked**	[0.65 0.00 0.35]	[1.00 0.00 0.00]	176.9	-189.3
4	**Ftpd_attacked_detector**	[0.40 0.12 0.48]	[0.93 0.07 0.00]	165.8	-173.8
5	**Httpd_hacked**	[0.33 0.10 0.57]	[0.67 0.19 0.14]	197.4	-206.4
6	**Ftpd_hacked**	[0.12 0.00 0.88]	[0.96 0.00 0.04]	204.8	-203.5
7	**Website_defaced**	[0.33 0.33 0.33]	[0.33 0.33 0.33]	80.4	-80.0
8	**Webserver_sniffer**	[0.00 0.50 0.50]	[0.33 0.33 0.34]	716.3	-715.1
9	**Webserver_sniffer_detector**	[0.34 0.33 0.33]	[1.00 0.00 0.00]	148.2	-185.4
10	**Webserver_DOS_1**	[0.33 0.33 0.33]	[1.00 0.00 0.00]	106.7	-106.1
11	**Webserver_DOS_2**	[0.34 0.33 0.33]	[1.00 0.00 0.00]	96.5	-96.0
12	**Network_shut_down**	[0.33 0.33 0.33]	[0.33 0.33 0.33]	80.4	-80.0
13	**Fileserver_hacked**	[1.00 0.00 0.00]	[0.35 0.34 0.31]	1065.5	-1049.2
14	**Fileserver_data_stolen_1**	[1.00 0.00 0.00]	[1.00 0.00 0.00]	94.4	-74.0
15	**Workstation_hacked**	[1.00 0.00 0.00]	[0.31 0.32 0.37]	1065.5	-1049.2
16	**Workstation_data_stolen_1**	[1.00 0.00 0.00]	[1.00 0.00 0.00]	94.4	-74.0
17	**Fileserver_data_stolen_2**	[0.33 0.33 0.33]	[0.33 0.33 0.33]	80.4	-80.0
18	**Workstation_data_stolen_2**	[0.33 0.33 0.33]	[0.33 0.33 0.33]	80.4	-80.0

In each state the actors have a choice of actions from the finite total set of actions. For example, in the Normal Operation state, the attacker has the options of attack-http, attack-ftp and nothing.

What do we want to address?

- A priori payoffs result in limited applicability of results.

What do we want to address?

- A priori payoffs result in limited applicability of results.
- In real world scenarios, the payoffs are determined by who is participating, not only the innate rules.

What do we want to address?

- A priori payoffs result in limited applicability of results.
- In real world scenarios, the payoffs are determined by who is participating, not only the innate rules.
- Skilled vs. unskilled adversary

What do we want to address?

- A priori payoffs result in limited applicability of results.
- In real world scenarios, the payoffs are determined by who is participating, not only the innate rules.
- Skilled vs. unskilled adversary
- What is the motivation of our adversary?

What do we want to address?

- A priori payoffs result in limited applicability of results.
- In real world scenarios, the payoffs are determined by who is participating, not only the innate rules.
- Skilled vs. unskilled adversary
- What is the motivation of our adversary?
- If we assume a priori the attackers utilities, then our predictions about their actions will be flawed.

- How can we formulate a game which allows us to discover this information about adversaries through play, can serve as a model for real world scenarios, and doesn't require a priori payoff information?

- How can we formulate a game which allows us to discover this information about adversaries through play, can serve as a model for real world scenarios, and doesn't require a priori payoff information?
- Having created such a game, can we create a proof of concept bot which improves its play with a simple opponent model eliciting a notion of opponent secret information from their play?

von Neumann's Betting Game (1944)

- The von Neumann betting game is a simplification of poker.
- It is a two-player game structured as follows:
  - Each player antes 1 unit to the pot
  - Each player receives a hand x: $x \in [0, 1]$.
  - Player 1 may either bet $B > 0$ or check
  - Player 2 may either call or fold
- The player with the best card wins
  - Unless a player folded. In which case the player who did not wins.

High Card

- High Card is a multiplayer adaptation of the von Neumann betting game, which hews closer to real poker.

High Card

- High Card is a multiplayer adaptation of the von Neumann betting game, which hews closer to real poker.
  - Each player is dealt a card from a deck without replacement.

High Card

- High Card is a multiplayer adaptation of the von Neumann betting game, which hews closer to real poker.
  - Each player is dealt a card from a deck without replacement.
  - Some players are forced to ante an amount of resources. Normal poker betting proceeds.

High Card

- High Card is a multiplayer adaptation of the von Neumann betting game, which hews closer to real poker.
  - Each player is dealt a card from a deck without replacement.
  - Some players are forced to ante an amount of resources. Normal poker betting proceeds.
  - The player with the best card wins.

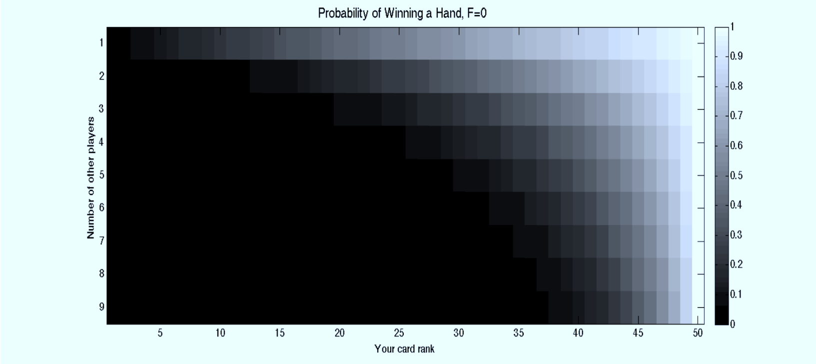
High Card

- High Card is a multiplayer adaptation of the von Neumann betting game, which hews closer to real poker.
  - Each player is dealt a card from a deck without replacement.
  - Some players are forced to ante an amount of resources. Normal poker betting proceeds.
  - The player with the best card wins.
- Conceptually, a game of High Card (one deal of cards) is like a round of betting in poker.

High Card

- High Card is a multiplayer adaptation of the von Neumann betting game, which hews closer to real poker.
  - Each player is dealt a card from a deck without replacement.
  - Some players are forced to ante an amount of resources. Normal poker betting proceeds.
  - The player with the best card wins.
- Conceptually, a game of High Card (one deal of cards) is like a round of betting in poker.
- A high card tournament involves a series of High Card games.
  - A set, equal number of resources is given to each player at the start of the tournament, and a final number of players M is chosen.
  - The tournament continues until M or fewer players remain with resources.

High Card Win Chance Map



- To model different player strategies we tested a number of different risk profiles.
- For these equations, W is the player's chance of Winning, P is the size of the Pot, B is the size of the Bet they would make. The ρ parameters are parameters of the function inherent to the player. Each function U accepts the arguments (P,B,W).

- To model different player strategies we tested a number of different risk profiles.
- For these equations, W is the player's chance of Winning, P is the size of the Pot, B is the size of the Bet they would make. The ρ parameters are parameters of the function inherent to the player. Each function U accepts the arguments (P,B,W).
    - linear: $U_l = W * P - (1 - W) * B$

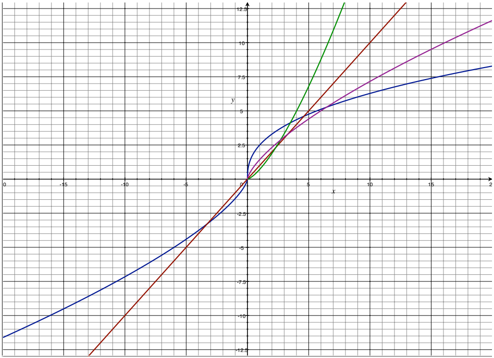
- To model different player strategies we tested a number of different risk profiles.
- For these equations, W is the player's chance of Winning, P is the size of the Pot, B is the size of the Bet they would make. The ρ parameters are parameters of the function inherent to the player. Each function U accepts the arguments (P,B,W).
  - linear: $U_l = W * P - (1 - W) * B$
  - superlinear: $U_{sl(\rho)} = W * \frac{P^{1+\rho}}{1+\rho} - (1 - W) * \frac{B^{1+\rho}}{1+\rho}$

- To model different player strategies we tested a number of different risk profiles.
- For these equations, W is the player's chance of Winning, P is the size of the Pot, B is the size of the Bet they would make. The ρ parameters are parameters of the function inherent to the player. Each function U accepts the arguments (P,B,W).
  - linear: $U_l = W * P - (1 - W) * B$
  - superlinear: $U_{sl(\rho)} = W * \frac{P^{1+\rho}}{1+\rho} - (1 - W) * \frac{B^{1+\rho}}{1+\rho}$
  - sublinear: $U_{bl(\rho)} = W * \frac{P^{1-\rho}}{1-\rho} - (1 - W) * \frac{B^{1-\rho}}{1-\rho}$

- To model different player strategies we tested a number of different risk profiles.
- For these equations, W is the player's chance of Winning, P is the size of the Pot, B is the size of the Bet they would make. The ρ parameters are parameters of the function inherent to the player. Each function U accepts the arguments (P,B,W).
  - linear: $U_l = W * P - (1 - W) * B$
  - superlinear: $U_{sl(\rho)} = W * \frac{P^{1+\rho}}{1+\rho} - (1 - W) * \frac{B^{1+\rho}}{1+\rho}$
  - sublinear: $U_{bl(\rho)} = W * \frac{P^{1-\rho}}{1-\rho} - (1 - W) * \frac{B^{1-\rho}}{1-\rho}$
  - prospect: $U_{p(\rho_a,\rho_b)} = W * \frac{P^{1-\rho_a}}{1-\rho_a} - (1 - W) * \frac{B^{1-\rho_b}}{1-\rho_b}$

- To model different player strategies we tested a number of different risk profiles.
- For these equations, W is the player's chance of Winning, P is the size of the Pot, B is the size of the Bet they would make. The ρ parameters are parameters of the function inherent to the player. Each function U accepts the arguments (P,B,W).
  - linear: $U_l = W * P - (1 - W) * B$
  - superlinear: $U_{sl(\rho)} = W * \frac{P^{1+\rho}}{1+\rho} - (1 - W) * \frac{B^{1+\rho}}{1+\rho}$
  - sublinear: $U_{bl(\rho)} = W * \frac{P^{1-\rho}}{1-\rho} - (1 - W) * \frac{B^{1-\rho}}{1-\rho}$
  - prospect: $U_{p(\rho_a,\rho_b)} = W * \frac{P^{1-\rho_a}}{1-\rho_a} - (1 - W) * \frac{B^{1-\rho_b}}{1-\rho_b}$
  - cumulative prospect:
    $U_{cp(\rho_a,\rho_b)} = f(W) * \frac{P^{1-\rho_a}}{1-\rho_a} - f(1 - W) * \frac{B^{1-\rho_b}}{1-\rho_b}$

Utility Function Plot



- *Linear*: Red
- *Sublinear*$_{(.7)}$: Purple
- *Superlinear*$_{(.4)}$: Green
- *Prospect*$_{(.6,.3)}$: Blue

Simple Bots

- We created a bot which can operate using an arbitrary utility function. Tested a number of these bots using different functions.

Simple Bots

- We created a bot which can operate using an arbitrary utility function. Tested a number of these bots using different functions.
- They estimate their chance of winning using Hypergeometric

Simple Bots

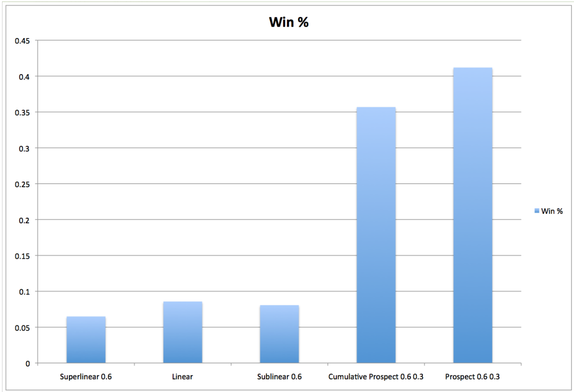
- We created a bot which can operate using an arbitrary utility function. Tested a number of these bots using different functions.
- They estimate their chance of winning using Hypergeometric
- The bots will play to maximize their bet while maintaining positive expected utility.

Simple Bots

- We created a bot which can operate using an arbitrary utility function. Tested a number of these bots using different functions.
- They estimate their chance of winning using Hypergeometric
- The bots will play to maximize their bet while maintaining positive expected utility.
- We tested a number of utility functions against each other in order to select the most promising for this particular game.

Utility Function Selection

Prospect Utility is appealing both because of its good performance in this game, and because it is empirically based on human behavior (Kahneman 1979).



- We now have a formulation of a game to play, as well as a set of opponents to play against.

- We now have a formulation of a game to play, as well as a set of opponents to play against.
- We want then to develop techniques to improve our play against those opponents.

- We now have a formulation of a game to play, as well as a set of opponents to play against.
- We want then to develop techniques to improve our play against those opponents.
- By observing past play, we should then be able to elicit information about the secret information those opponents currently hold.

Opponent Modeling

Using an estimate of our opponent's secret information we can generate our updated estimate of our chance to win.

- C_{mod} is the card the modeling bot has.
- N is the total number of players who haven't folded.
- C_i is the card player i has.
- max_i is the highest card observed for player i in a situation considered relevant.

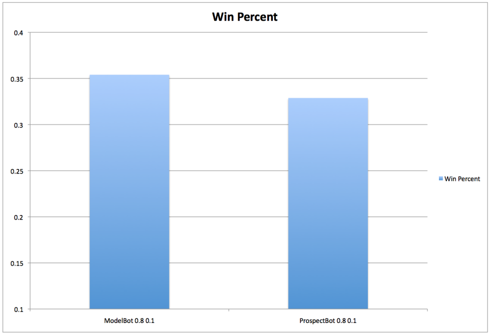
$$W = \prod_{i=0}^{N} P(C_{mod} > C_i)$$

$$P(C_{mod} > C_i) = \begin{cases} \frac{\binom{C_{mod}-1}{1}}{\binom{S}{1}} & \text{if } max_i = \emptyset \\ 1 & \text{if } C_{mod} = C_i \text{ or } C_{mod} > max_i \\ \frac{C_{mod}-min_i}{max_i-min_i-1} & \text{if } C_{mod} < max_i \text{ and } C_{mod} > min_i \\ 0 & \text{if } C_{mod} < min_i \end{cases}$$

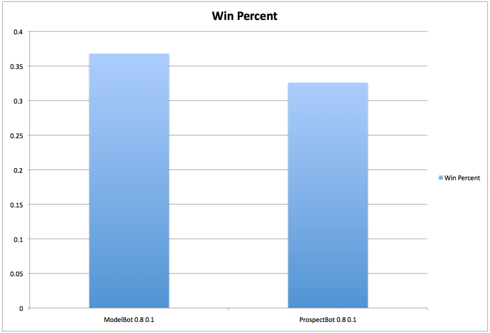
- The following results are a selection of the data collected from running our bot, and are indicative of its performance with various parameters of the Prospect Utility function.

- The following results are a selection of the data collected from running our bot, and are indicative of its performance with various parameters of the Prospect Utility function.
- The method of modeling which our bot utilized is very simple, so there is a lot of potential for better exploitation of the very limited information gathered.

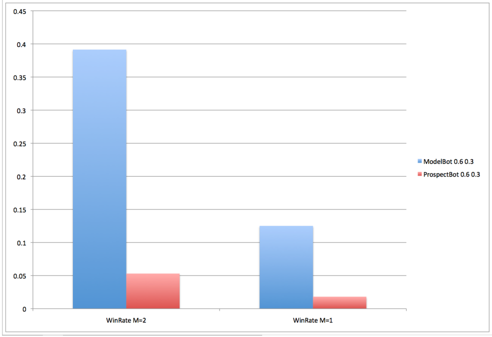
- 1000 Tournaments
- $M = 2$
- No data between tournaments.
- Against bots with same utility function



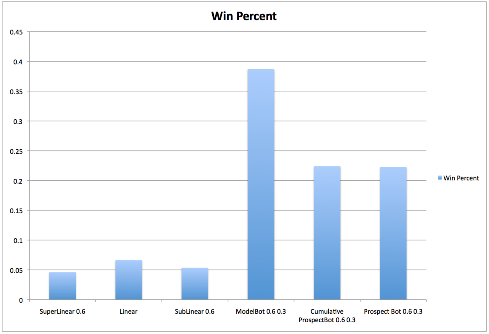
- 1000 Tournaments
- $M = 2$
- Data stored between tournaments.
- Against bots with same utility function



- 1000 Tournaments
- $M = 1$ and $M = 2$
- Data stored between tournaments. Data imported from previous tournaments.
- Against a variety of bots.



- 1000 Tournaments
- $M = 1$ and $M = 2$
- Data stored between tournaments. Data imported from previous tournaments.
- Against a variety of bots.



Future Work

- Building upon this work, we are interested in exploring what occurs if the actual outcomes of a game which can be represented ultimately as a normal form game are unknown.

Future Work

- Building upon this work, we are interested in exploring what occurs if the actual outcomes of a game which can be represented ultimately as a normal form game are unknown.
- This may be the case if the outcomes of the game are determined not just by the actions your opponents take, but by who they are, and what their objective is.

Future Work

- Building upon this work, we are interested in exploring what occurs if the actual outcomes of a game which can be represented ultimately as a normal form game are unknown.
- This may be the case if the outcomes of the game are determined not just by the actions your opponents take, but by who they are, and what their objective is.
- This ultimately, we believe, is a better model for an adversarial real world environment.

Future Work

- Building upon this work, we are interested in exploring what occurs if the actual outcomes of a game which can be represented ultimately as a normal form game are unknown.
- This may be the case if the outcomes of the game are determined not just by the actions your opponents take, but by who they are, and what their objective is.
- This ultimately, we believe, is a better model for an adversarial real world environment.
- We may even be in the situation of being an external observer watching a game. If we are only able to observe actions, but not payoffs, can we elicit what the game is, and having done that find out who the players are?