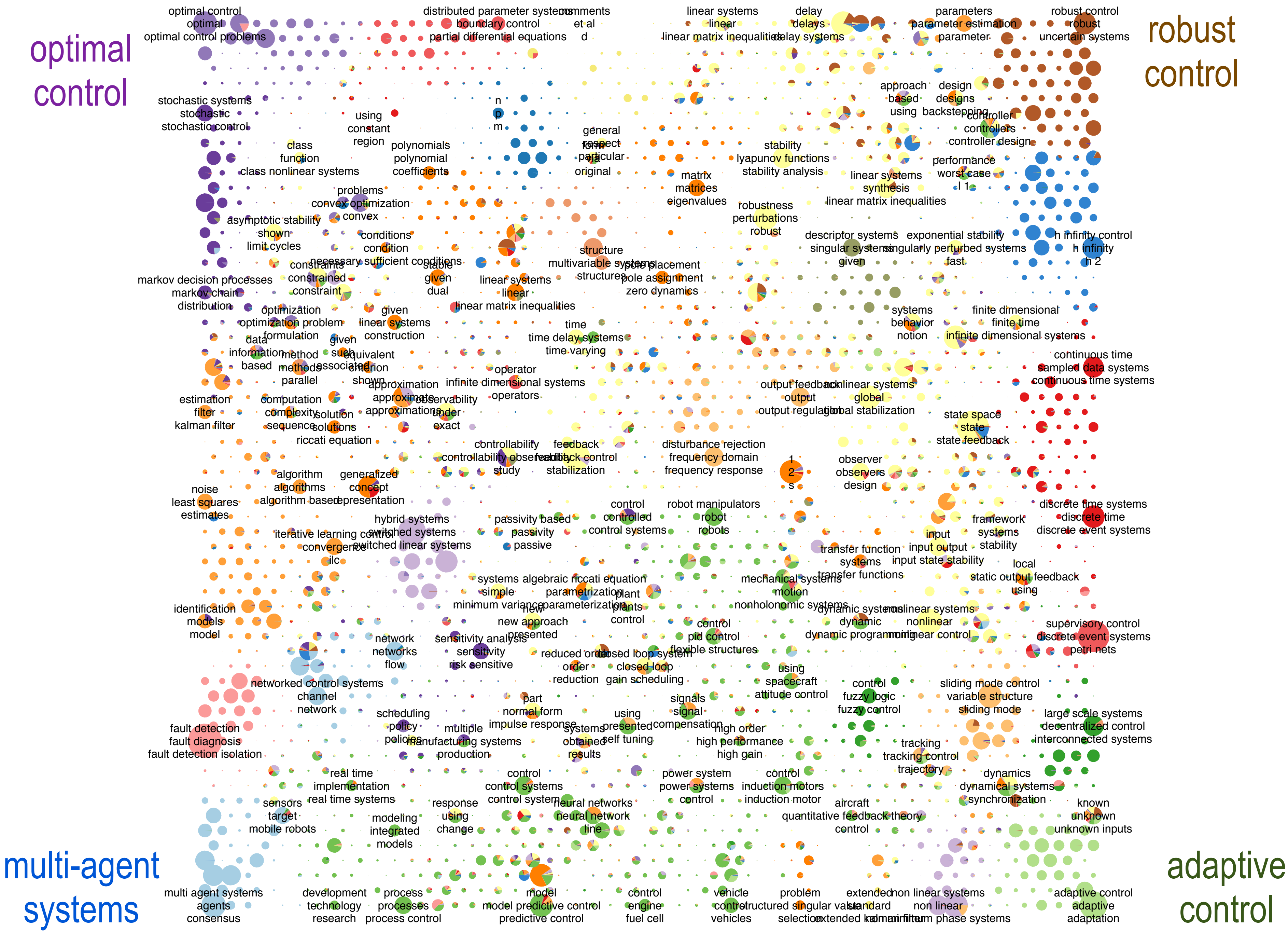
Randomized Methods for Network Security Games

João P. Hespanha

Joint work with: S. Bopardikar, M. Prandini, A. Borri, M. D. Di Benedetto

The Geography of Control

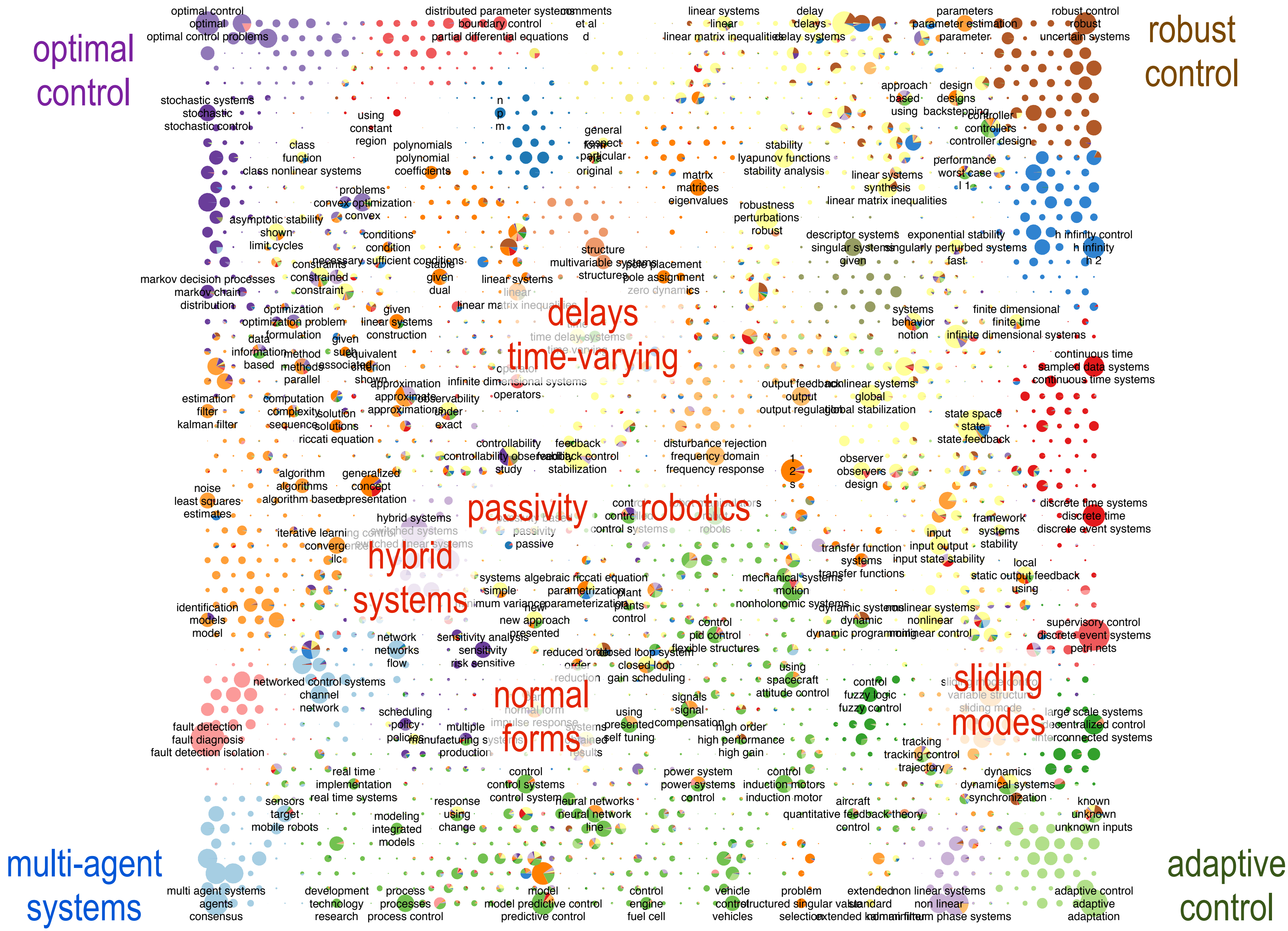
UC SANTA BARBARA engineering



In collaboration with Stacy Rebich Hespanha

The Geography of Control

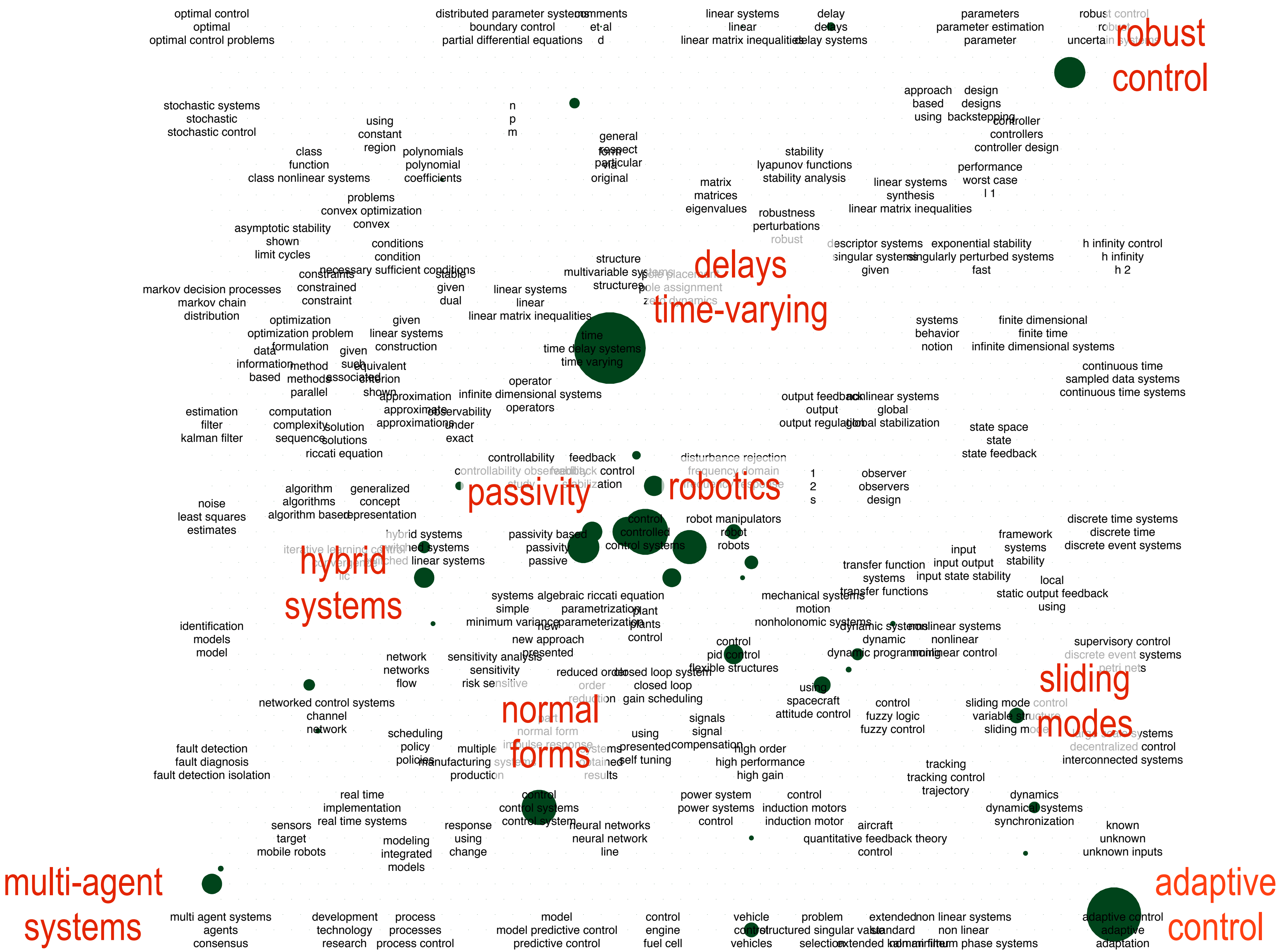
UC SANTA BARBARA
engineering



In collaboration with Stacy Rebich Hespanha

The Geography of Mark Spong

UC SANTA BARBARA
engineering



In collaboration with Stacy Rebich Hespanha

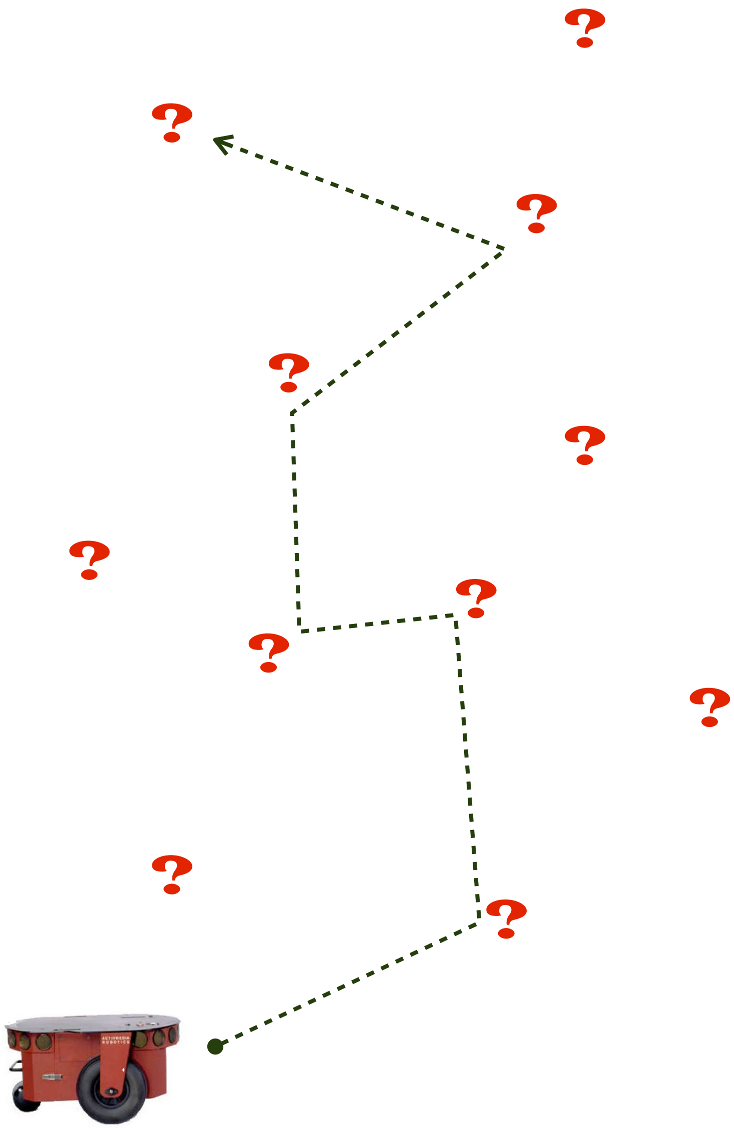
The Plan...

UC SANTA BARBARA engineering

- Motivation for large-scale games
- Zero-Sum Games, Security Policies
- Randomized Algorithms for Games (SSP Algorithm)

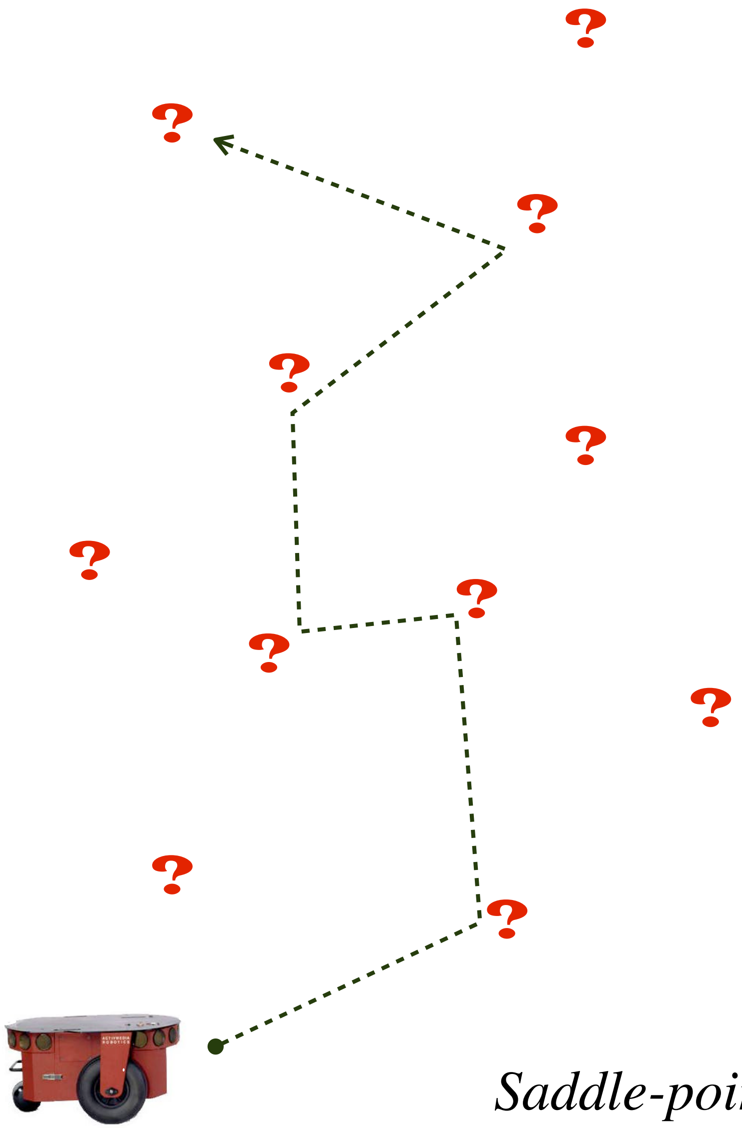
- Case Study in Network Security (time permiting...)

Path Planning Games: Hide & Seek



What is the "best" way to hide a treasure among N hiding places in the plane ?

Path Planning Games: Hide & Seek



What is the "best" way to hide a treasure among N hiding places in the plane ?

Formulate problem as *game*
- P1 selects hiding place (N locations)
- P2 selects robot path (**N! paths**)

Zero-sum game
- P1 wants to maximize time to reach treasure
- P2 has opposite objective

Saddle-point is a mixed policy

P1 places treasure based on "optimal" distribution
P2 selects path based on "optimal" distribution

Which???

Network Security

Cybaware

Cyber Situation Awareness ARO MURI:

Protecting the cyber infrastructure that supports "missions"

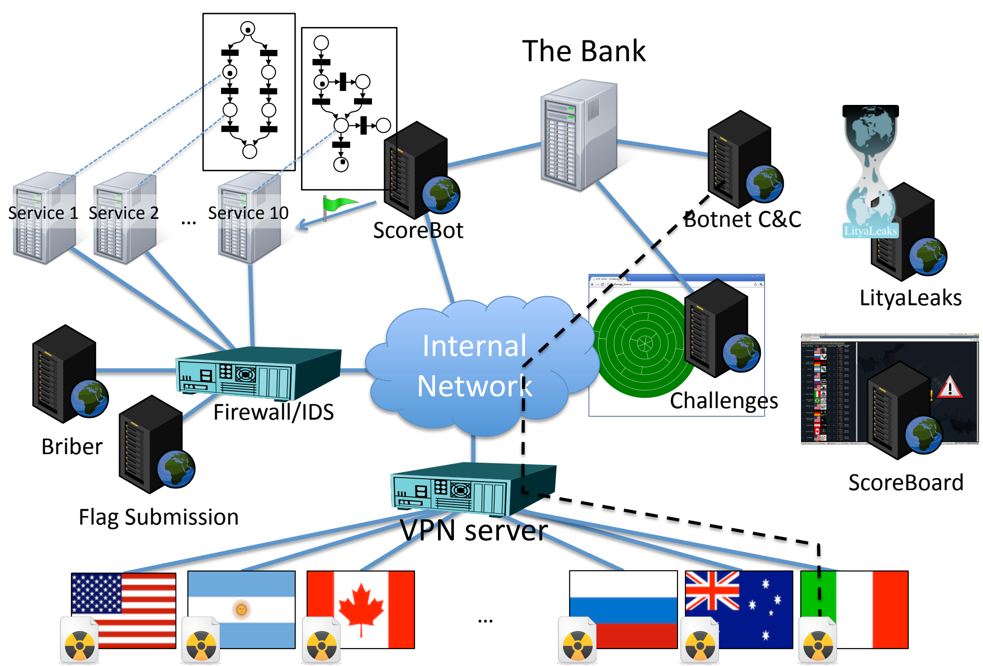
processing an online purchase

monitoring a critical infrastructure

managing the admissions process at a university

Cyber Awareness Questions:

- What is in the impact of a cyber-asset vulnerability (known or unknown)?
- What is in the impact of a particular counter measure (e.g., firewall control)?



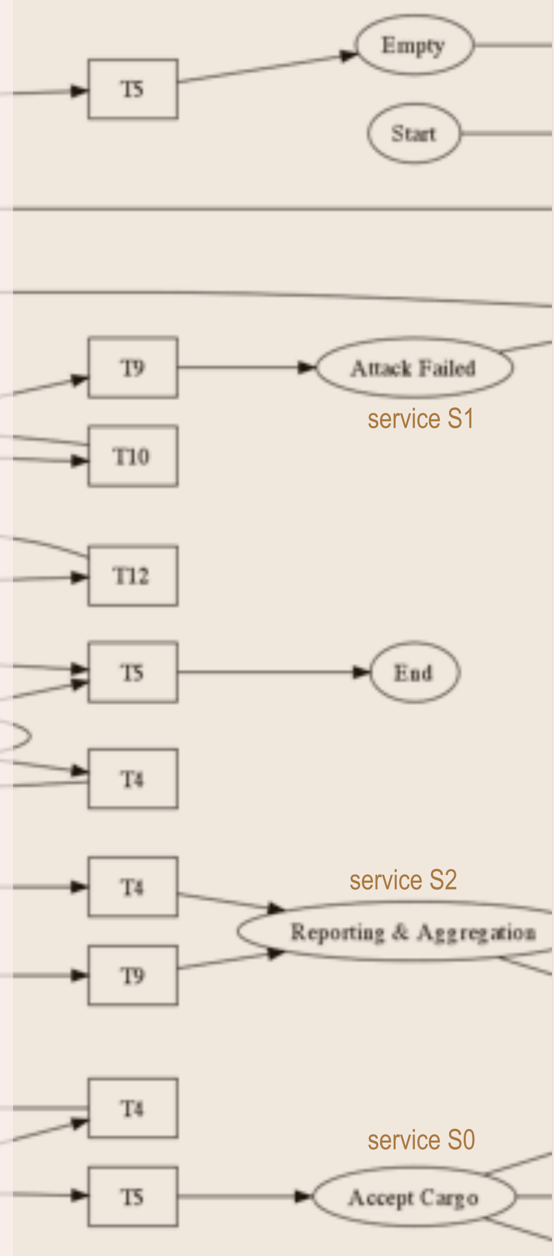
International Capture the Flag (iCTF):

- Distributed, wide-area security exercise to test the security skills of the participants
- Held yearly since 2003, under the organization of Prof. Giovanni Vigna @ UCSB
- 2010 edition involved 72 teams from around the world, over 900 participants

2010 iCTF Game

Key ingredients

1. *Game objective & Actors*
   - defender runs web server and collects revenue (when firewalls open) and bribes (when firewalls closed)
   - attacker collects revenue by compromising active services
2. *Players' Actions*
   - defender controls firewall
   - attacker explores service vulnerabilities
3. *Information Structure*
   - defender knows current status of several missions and services active
   - attacker has access to full state information (option I) or partial state information (option II)

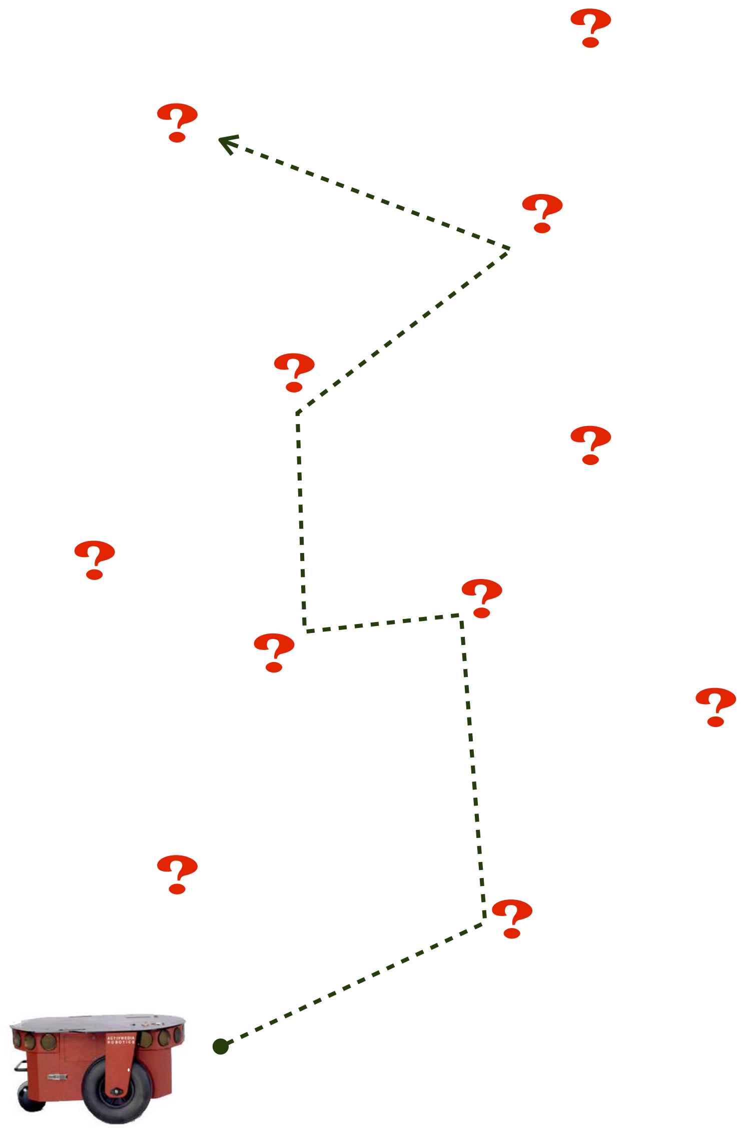


Problem statistics:

- over 7800 distinct mission states (defender observations)
- over 2500 distinct observations available to the attacker
- defender can choose among about 10^{2527} distinct policies
- attacker can choose among 10^{756} – 10^{2616} distinct policies, depending on attacker's level of expertise

Randomized Search (H&S)

UC SANTA BARBARA engineering



What is the "best" way to hide a treasure among N hiding places in the plane ?

- P1 selects robot path (**N! paths**)
- P2 selects hiding place (N locations)

Randomized algorithms:
When the exhaustive exploration of a combinatorial decision tree is not possible…
explore a random subset of it

Motivation for this work:
How can a player "protect" herself from an opponent engaged in random exploration?

Zero-Sum Matrix Games

Two players:
- P1 - minimizer
- P2 - maximizer

Each player selects a policy:
- S1 - set of available policies for P1
- S2 - set of available policies for P2

All possible game outcomes can be encoded in a matrix (2D-array), jointly indexed by the actions of the players

game outcome when
P1 selects policy i
P2 selects policy j

$$\begin{bmatrix} & \vdots & \\ \cdots & a_{ij} & \cdots \\ & \vdots & \end{bmatrix}$$

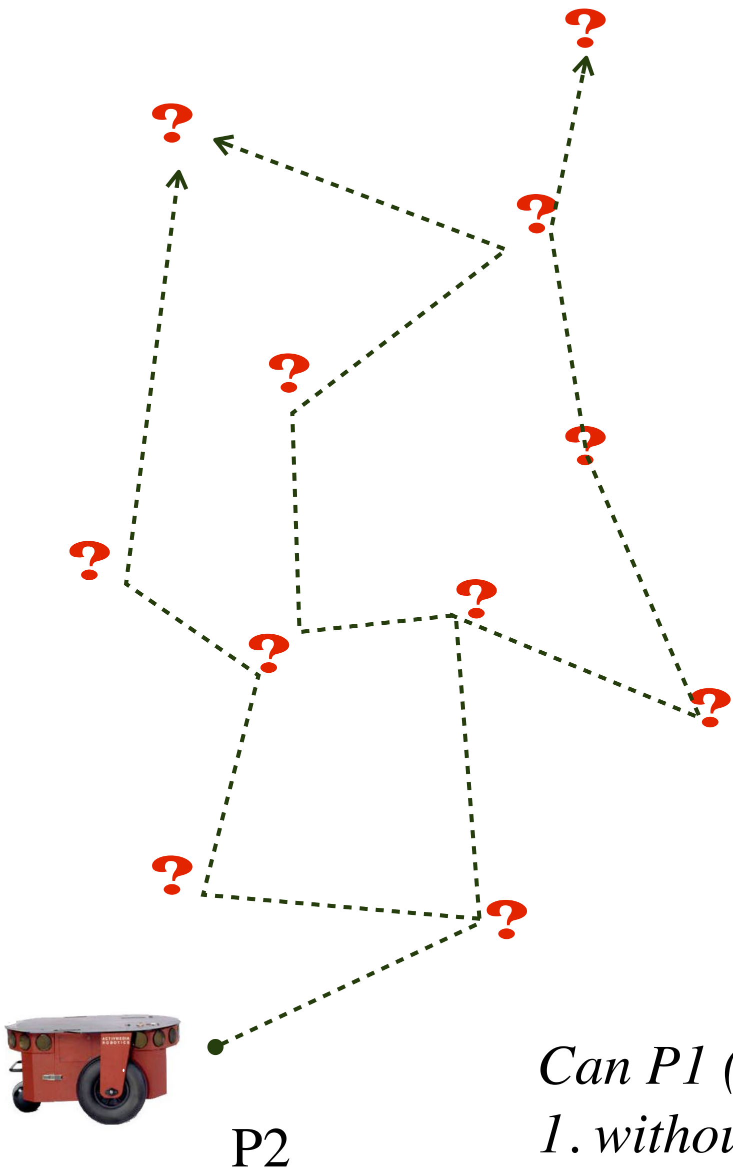
P1's policy

P2's policy

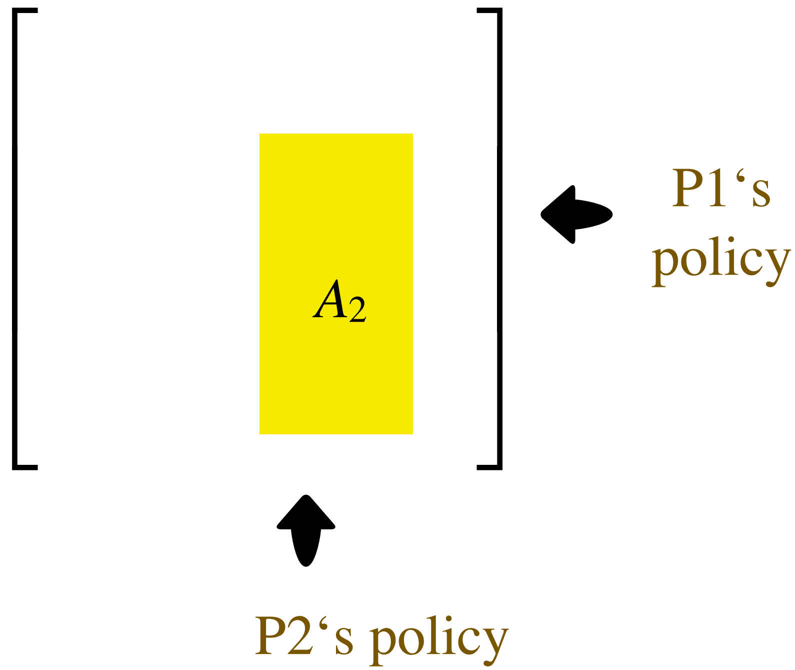
We are interested in games in which, at least one (possibly both) dimensions of this matrix are very large.

Mixed security level for P1 (minimizer): $V := \min_y \max_z \mathrm{E}[a_{ij}] = \min_y \max_z y' A z$

Randomized Search in Matrix Games



Suppose P2 (searcher) only considers an (unknown) random subset of the $N!$ paths and (somehow) selects a policy based on those...

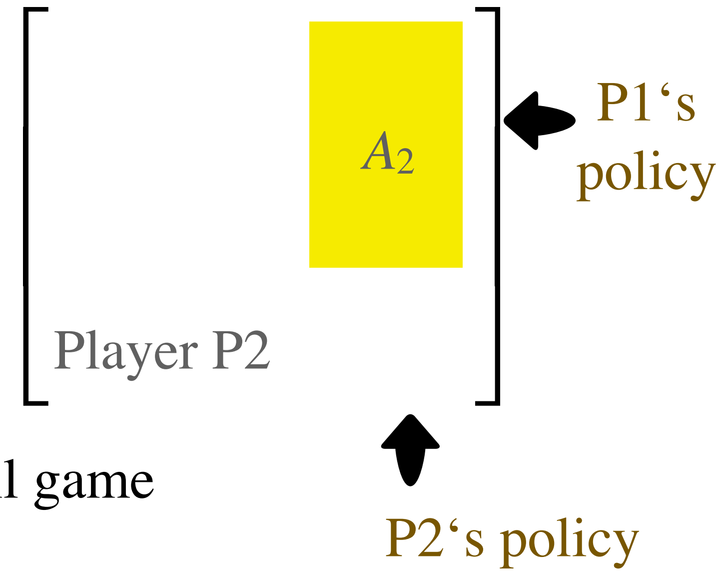


Can P1 (hider) guarantee some minimum search time

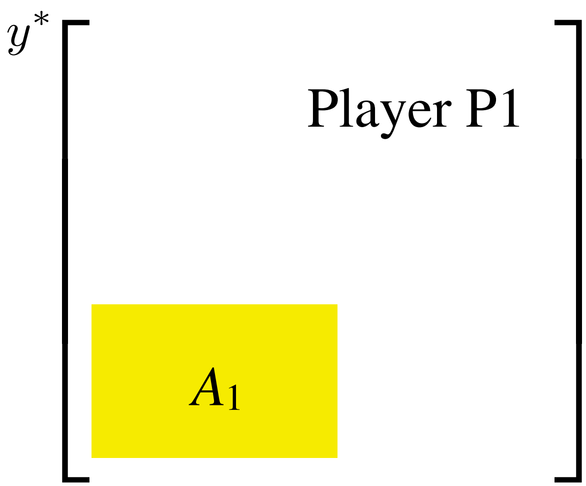
1. *without knowing which random subset was chosen*
2. *with a small computational effort ?*

Sampled-Security Policy Algorithm (SSP)

Suppose P2 only considers an (unknown) random subset of its policies to compute a (mixed) policy z^*...



1. Player P1 randomly selects a submatrix of the overall game



2. P1 solves the corresponding subgame (as if it was the whole game) and computes
   - mixed security level V_1
   - corresponding security policy y^*
3. P1 plays y^* against P2's policy z^*

in very large games, submatrix A_1 will likely not overlap the matrix A_2 used by P2

Sampled-Security Policy Algorithm (SSP)

Suppose P2 only considers an (unknown) random subset of its policies to compute a (mixed) policy z^*...

A_2

Player P2

1. Player P1 randomly selects a submatrix of the overall game

y^*

Player P1

A_1

in very large games, submatrix A_1 will likely not overlap the matrix A_2 used by P2

2. P1 solves the corresponding subgame
   - mixed security level V_1
   - corresponding security policy y^*
3. P1 plays y^* against P2's policy z^*

Because of independent subsampling, a player can now be unpleasantly surprised:

$$\underbrace{E_{y^*,z^*}[a_{ij}] > V_1}$$

outcome larger than minimizer expected based on its submatrix A_1

Security with High Probability

Probabilistic notion of security:

- probability of (unpleasant) surprises should be below a pre-specified bound
- with more computational power, one can demand lower prob. of surprise

Definition: The SSP algorithm is $\epsilon-$*secure for P1 (minimizer) with confidence 1*$-\delta$ if

$$\mathrm{P}(\underbrace{\mathrm{E}_{y*,z*}[a_{ij}] > V_1 + \epsilon}_{\substack{\text{outcome larger than} \\ \text{what P1 expected} \\ \text{(by more than } \epsilon\text{)}}}) \leqslant \delta$$

"Surprise" can arise because:

- our policy y^* is actually bad
- our security value V_1 is overly optimistic
- P2 was lucky in the selection of z^*

Security with High Probability

Probabilistic notion of security:

- probability of (unpleasant) surprises should be below a pre-specified bound
- with more computational power, one can demand lower prob. of surprise

Definition: The SSP algorithm is ϵ–*secure for P1 (minimizer) with confidence 1*–δ if

$$P(\underbrace{E_{y*,z*}[a_{ij}] > V_1 + \epsilon}_{\text{outcome larger than what P1 expected (by more than } \epsilon)}) \leqslant \delta$$

"Surprise" can arise because:

- our policy y^* is actually bad
- our security value V_1 is overly optimistic
- P2 was lucky in the selection of z^*

avoidable with very high probability

not so easily under our control

Security with High Probability

Probabilistic notion of security:

- probability of (unpleasant) surprises should be below a pre-specified bound
- with more computational power, one can demand lower prob. of surprise

Definition: The SSP algorithm is ϵ−*secure for P1 (minimizer) with confidence 1*−δ if

$$\mathrm{P}(\underbrace{\mathrm{E}_{y*,z*}[a_{ij}] > V_1 + \epsilon}_{\text{outcome larger than P1 expected (by more than } \epsilon)}) \leqslant \delta$$

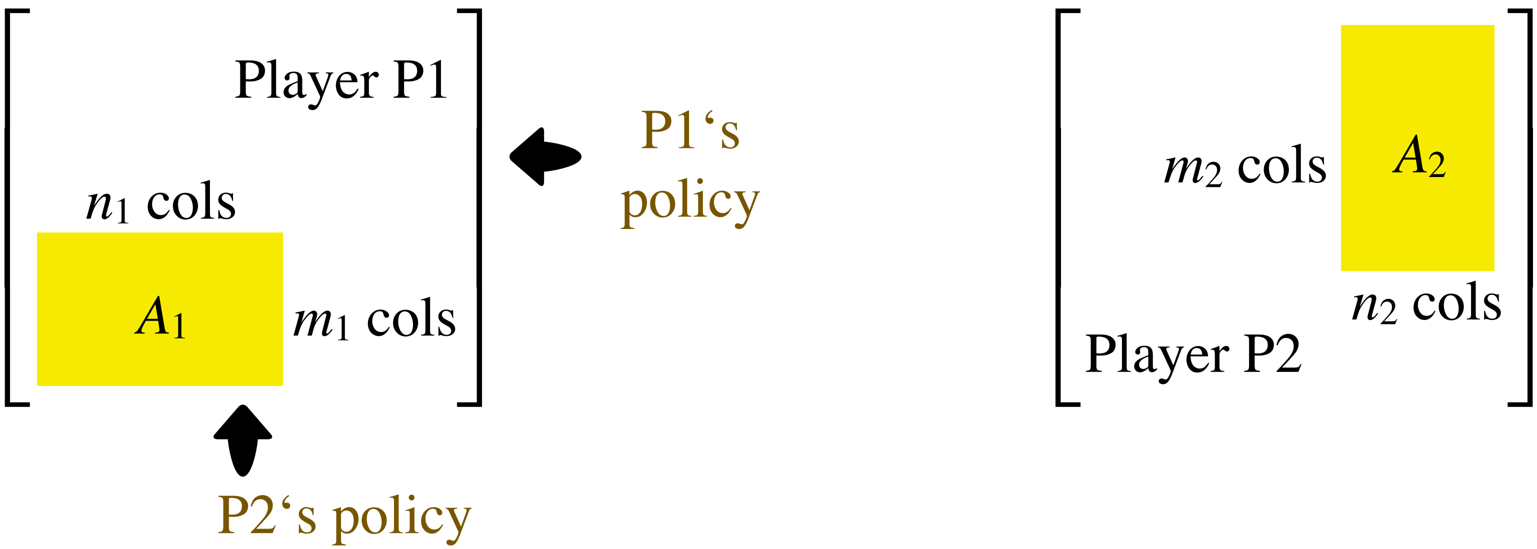
Definition: A particular policy y^* and value V_1 is ϵ−*secure for P1 (minimizer) with confidence 1*−δ if

$$\mathrm{P}(\mathrm{E}_{y*,z*}[a_{ij}] > V_1 + \epsilon \mid y^*, V_1) \leqslant \delta$$

"Surprise" only because:

- P2 was lucky in the selection of z^*

SSP Key Result



Theorem: The SSP algorithm is $\epsilon = 0$ – *secure for P1 (minimizer) with confidence 1–δ*, for

$$\delta = \frac{m_1 n_2}{n_1}$$

Conversely, to obtain desired confidence level δ, suffices to select

$$\frac{n_1}{m_1} \geqslant \frac{n_2}{\delta} > 1$$

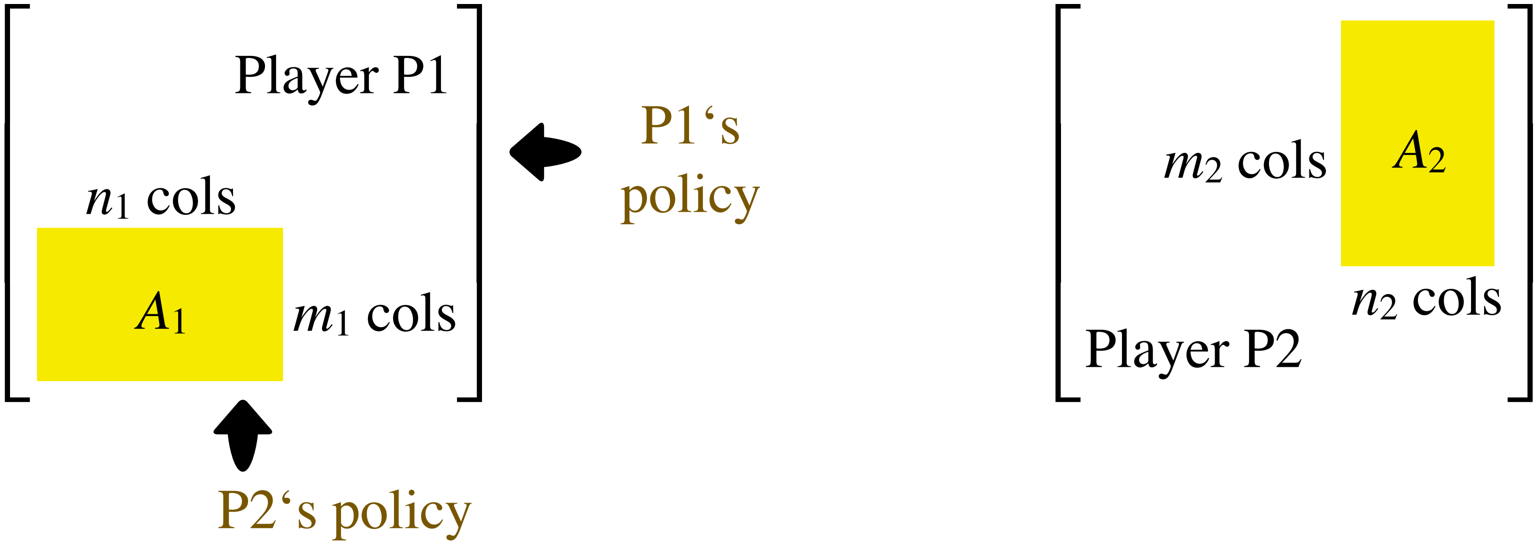
"fat" sampling for A_1

$\Downarrow$

test more options for opponent than own (by appropriate ratio)

SSP Key Result

UC SANTA BARBARA engineering



Theorem: The SSP algorithm is $\epsilon = 0$ – *secure for P1 (minimizer) with confidence* $1-\delta$, for $m_1 n_2$

Conversely, to obtain desired

$$\frac{n_1}{m_1} \geqslant \frac{n_2}{\delta} > 1$$

Game-independent bounds

- valid for any game
- independent of the size of the game

Bounds on *relative computation*

- required size of my sample depends on size of opponents' sample (n_2)
- the more I search for a good solution (large m_1), the more options need to consider for opponent (large n_1)

Security with High Probability (recall)

Definition: The SSP algorithm is $\epsilon-$*secure for P1 (minimizer) with confidence 1−δ* if

$$\mathrm{P}(\underbrace{\mathrm{E}_{y*,z*}[a_{ij}] > V_1 + \epsilon}_{\text{outcome larger than P1 expected (by more than } \epsilon\text{)}}) \leqslant \delta$$

"Surprise" can arise because:

- our policy y^* is actually bad
- our security value V_1 is overly optimistic
- P2 was lucky in the selection of z^*

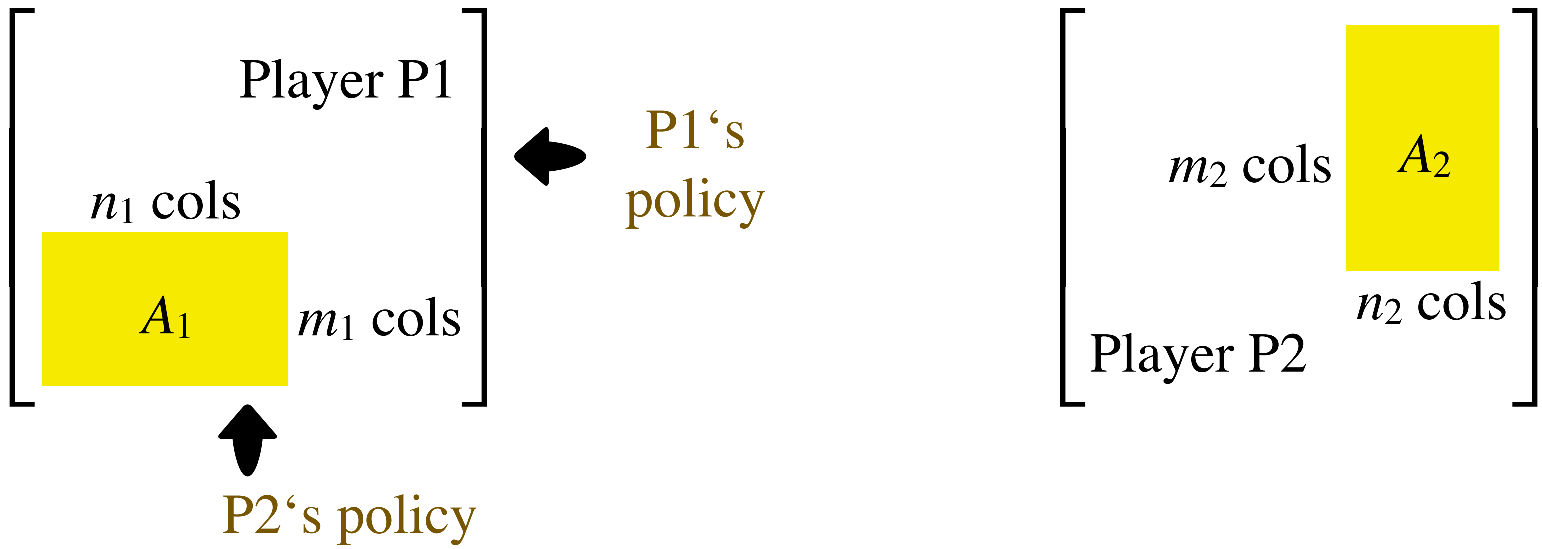
Definition: A particular policy y^* and value V_1 is $\epsilon-$*secure for P1 (minimizer) with confidence 1−δ* if

$$\mathrm{P}(\mathrm{E}_{y*,z*}[a_{ij}] > V_1 + \epsilon \mid y^*, V_1) \leqslant \delta$$

"Surprise" only because:

- P2 was lucky in the selection of z^*

SSP Key Result (take 2)



Theorem: With probability larger than $1-\beta$, the SSP policy y^* and value V_1 are $\epsilon = 0$ – *secure for P1 (minimizer) with confidence* $1-\delta$, for

$$n_1 \geqslant \Big(m_1 + \underbrace{\sqrt{m_1 \ln \frac{1}{\beta^2}}}_{\text{additional term}}\Big)\frac{n_2}{\delta}$$

"Surprise" can arise because:

- our policy y^* is actually bad
- our security value V_1 is overly optimistic
- P2 was lucky in the selection of z^*

only with probability β (can be made extremely small ~ 10^{-9} with small computational cost) [for the first two items]

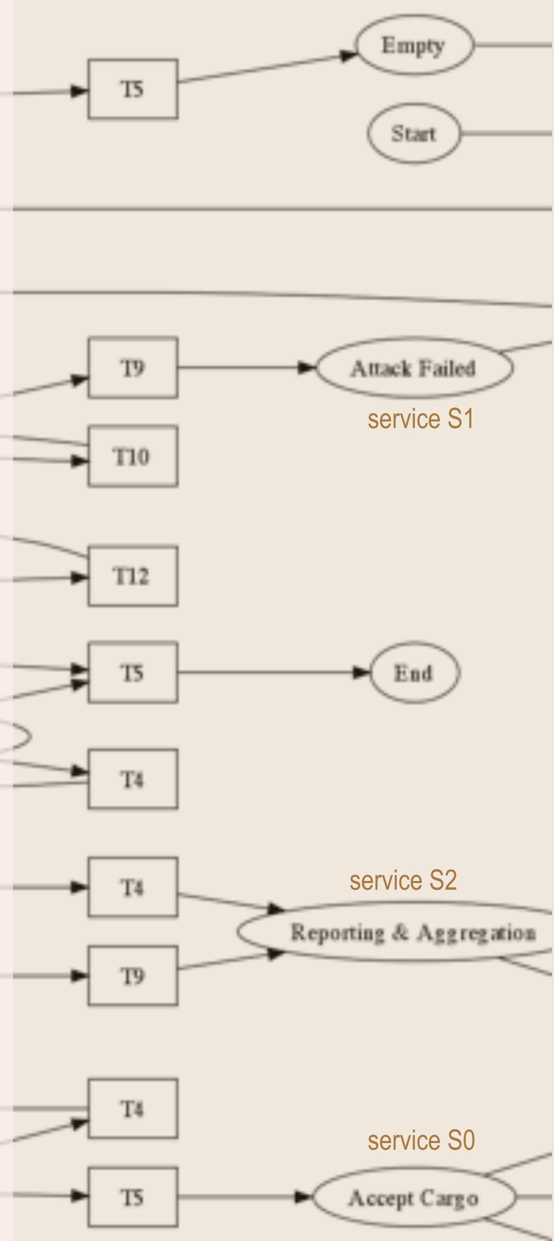
with probability δ [for the third item]

2010 iCTF Game

UC SANTA BARBARA engineering

Key ingredients

1. *Game objective & Actors*
   - defender runs web server and collects revenue (when firewalls open) and bribes (when firewalls closed)
   - attacker collects revenue by compromising active services
2. *Players' Actions*
   - defender controls firewall
   - attacker explores service vulnerabilities
3. *Information Structure*
   - defender knows current status of several missions and services active
   - attacker has access to full state information (option I) or partial state information (option II)



Problem statistics:

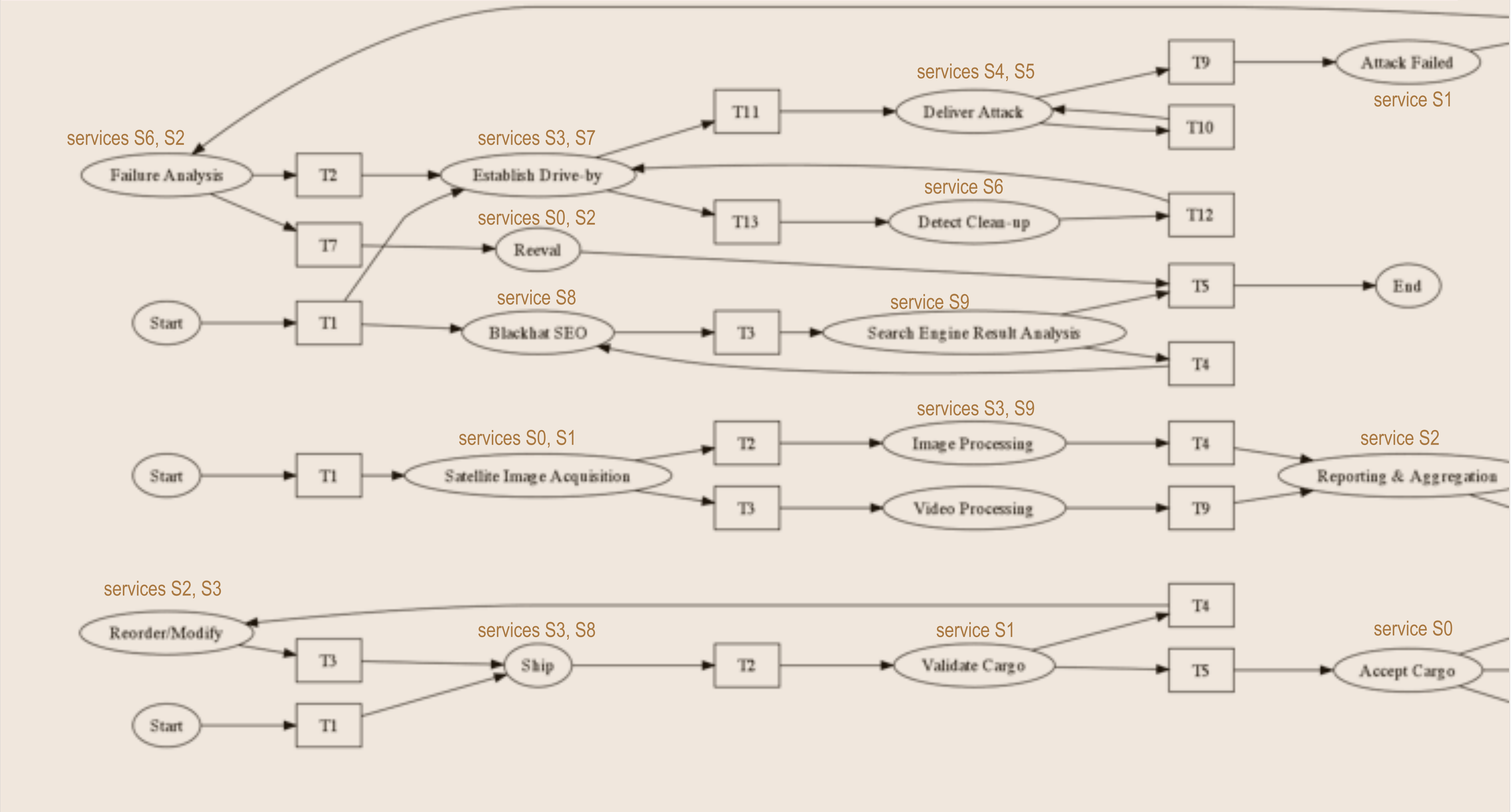
- over 7800 distinct mission states (defender observations)
- over 2500 distinct observations available to the attacker
- defender can choose among about 10^{2527} distinct policies
- attacker can choose among 10^{756} – 10^{2616} distinct policies, depending on attacker's level of expertise

SSP-based Analysis

Baseline

- No attack
- Firewall always open
- All revenue from bot

Defender receives avg. 314 units for completion of 4 missions



SSP-based Analysis

Baseline

- No attack
- Firewall always open
- All revenue from bot

Defender receives avg. 314 units for completion of 4 missions

Attacks under Option I (ambiguous state information)

- Different levels of attacker sophistication
- Bribes not allowed

Increasing level of attacker sophistication

Level of attacker sophistication	# units received by Defender for 1 round of missions [Option I, no bribes]
no service vulnerable (baseline)	314
S2 (vulnerable to 38 teams)	240
S2, S6, S9 (vulnerable to at least 6 team)	79
S0, S2, S4, S6, S7, S8, S9 (vulnerable to at least 1 team)	11
all services vulnerable	11

Firewall closed whenever a vulnerable service is active

SSP-based Analysis

UC SANTA BARBARA engineering

Baseline

- No attack
- Firewall always open
- All revenue from bot

Defender receives avg. 314 units for completion of 4 missions

Attacks under Option II (full state information)

- Different levels of attacker sophistication
- Bribes allowed

Increasing level of attacker sophistication

Level of attacker sophistication	# units received by Litya for 1 round of missions [Option I, no bribes]	# units received by Litya for 1 round of missions [Option I, with bribes]	# units received by Litya for 1 round of missions [Option II, with bribes]
no service vulnerable (baseline)	314	314	314
S2 (vulnerable to 38 teams)	240	240	138
S2, S6, S9 (vulnerable to at least 6 team)	79	79	43
S0, S2, S4, S6, S7, S8, S9 (vulnerable to at least 1 team)	11	-738	-1327
all services vulnerable	11	-848	-1917

SSP-based Analysis

Baseline

- No a
- Fire
- All r

Attac

- Diff
- Bribes allowed

issions

- Provides Cyber-security office estimates of mission success
- Takes into account the effect of attacks & counter measures
- Responses as a function of attacker sophistication
- Ability to play what-if scenarios (vulnerabilities, information, etc.)

Increasing level of attacker sophistication

Level of attacker sophistication	# units received by Litya for 1 round of missions [Option I, no bribes]	# units received by Litya for 1 round of missions [Option I, with bribes]	# units received by Litya for 1 round of missions [Option II, with bribes]
no service vulnerable (baseline)	314	314	314
S2 (vulnerable to 38 teams)	240	240	138
S2, S6, S9 (vulnerable to at least 6 team)	79	79	43
S0, S2, S4, S6, S7, S8, S9 (vulnerable to at least 1 team)	11	-738	-1327
all services vulnerable	11	-848	-1917

Conclusions

Large matrix games are fun!

What I have covered:

- Basic probability guarantees of randomized sampling for games
- Case study in network security

What I have NOT covered:

- Can we determine the security level of an arbitrary policy obtained by a method other than SSP? Yes
- Mistmatch between the players' distributions
  - if we sample "better" than opponent, no change in results
  - if we sample "worse" than opponent, degradation in confidence level δ (but recoverable with more sampling)
- Can we improve upon these bounds? Unclear
- Multi-core parallel implementations
- Other applications...

Conclusions

Large matrix games are fun!

What I have covered:

- Basic probability guarantees of randomized sampling for games
- Case study in network security

What I have NOT covered:

- Can we determine the security level of an arbitrary policy obtained by a method other than SSP? Yes
- Mistmatch between the players' distributions
  - if we sample "better" than opponent, no change in results
  - if we sample "worse" than opponent, degradation in confidence level δ (but recoverable with more sampling)
- Can we improve upon these bounds? Unclear
- Multi-core parallel implementations
- Other applications...

Thank You!