IJCAI Tutorial

Solving Games with Complex Strategy Spaces

Part II: Integrating Learning with Game Theory for Strategic Decision Making

Fei Fang
Carnegie Mellon University
feifang@cmu.edu

Outline

- Games with Human Players for Real-world Applications
  - Wildlife Conservation
- End-to-End Learning and Decision Making in Games
  - A differentiable learning framework for learning game parameters
- Learning-Powered Strategy Computation in Large Games
  - Leveraging Deep Reinforcement Learning
- Other Applications and Summary

Societal Challenges

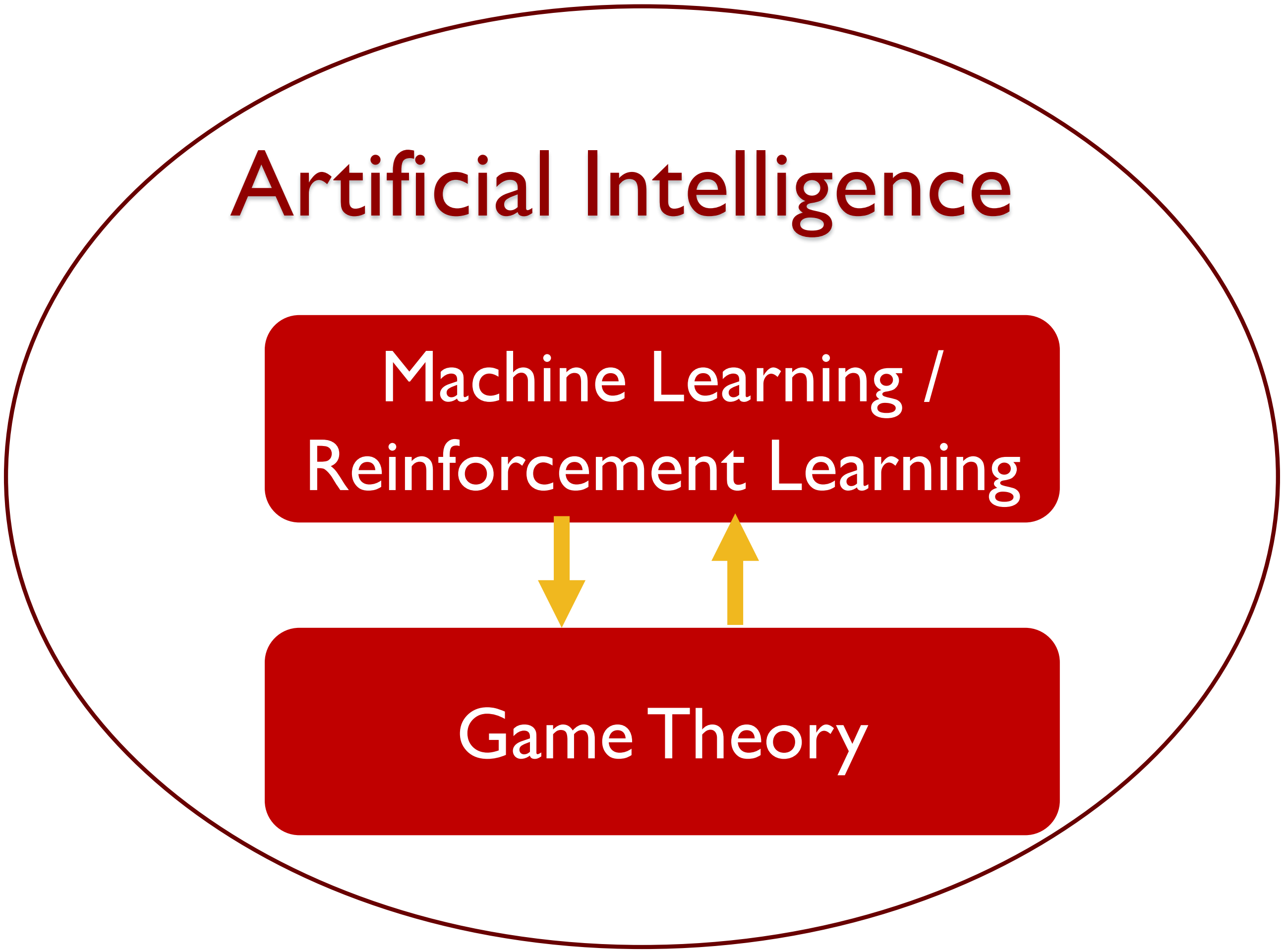
Security & Safety

Environmental Sustainability

Mobility

HAIL
EASY TAXI
lyft
Flywheel
OLA
GRABTAXI
Get Taxi
UBER
BLACKLANE LIMOUSINES
shuddle
Getaround
via
99TAXIS
Side.car

Solution Approaches



Recap: Security Games

- Strong Stackelberg Equilibrium
  - Defender: mixed strategy
  - Attacker: best response, break tie in favor of defender

Defender

Adversary

		Target #1	Target #2
55.6%	Target #1	5, -3	-1, 1
44.4%	Target #2	-5, 4	2, -1

Quiz

- How to get the defender's mixed strategy in SSE in this problem?

		Adversary	
		Target #1	Target #2
55.6%	Target #1	5, -3	-1, 1
44.4%	Target #2	-5, 4	2, -1

Quiz

- How to get the defender's mixed strategy in SSE in this problem?
  - AttEU1=$p*(-3)+(1-p)*4=p*1+(1-p)*(-1)$=AttEU2
  - Equilibrium: DefStrat=(0.556,0.444), AttStrat=(1,0)

Defender

Adversary

		Target #1	Target #2
55.6%	**Target #1**	5, -3	-1, 1
44.4%	**Target #2**	-5, 4	2, -1

Recap: SSE vs NE

- Zero-sum
  - SSE=NE=minimax=maximin
  - Approach 1: Single LP (minimax or maximin strategy)
  - Approach 2: Greedy allocation for security games

- General-sum
  - SSE$\geq$NE
  - Computing NE: PPAD Complete, LCP (linear complementarity problem) formulation, Gambit solver
  - Computing SSE
    - Approach 1: Multiple LPs (each solve a subproblem)
    - Approach 2: A single MILP that combines all the LPs
    - Approach 3: Extended greedy allocation algorithm $O(nlog\ n)$ for security games

Example: Protecting Staten Island Ferry

Optimal Patrol Strategy for Protecting Moving Targets with Multiple Mobile Resources. Fei Fang, Albert Xin Jiang, Milind Tambe. In AAMAS-13

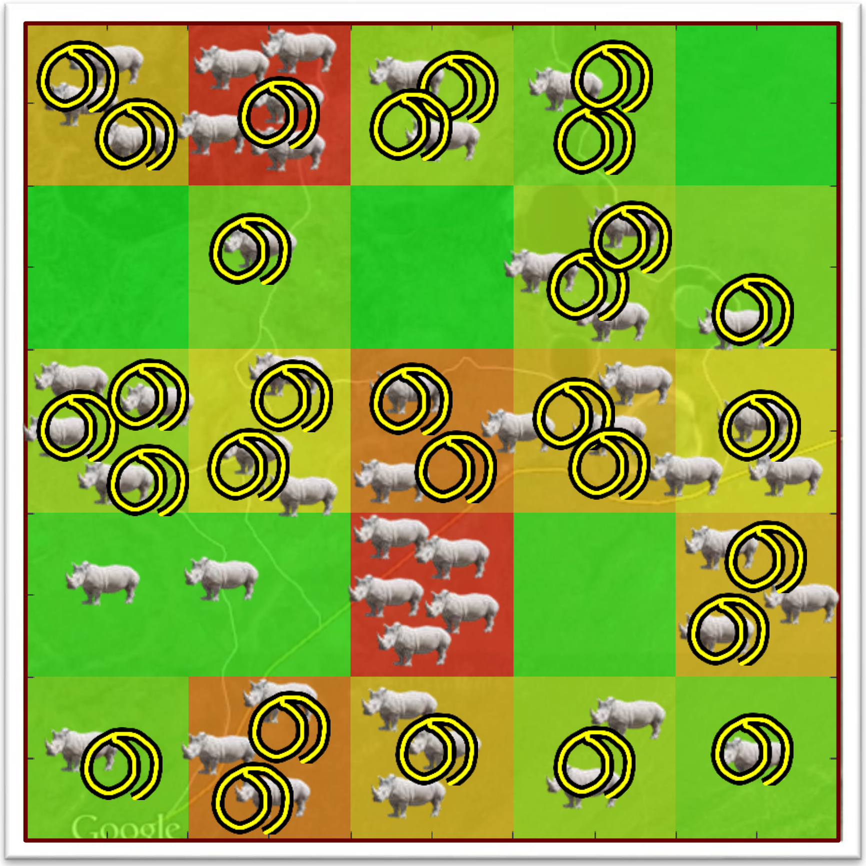
Outline

- Games with Human Players for Real-world Applications
  - Wildlife Conservation
- End-to-End Learning and Decision Making in Games
  - A differentiable learning framework for learning game parameters
- Learning-Powered Strategy Computation in Large Games
  - Leveraging Deep Reinforcement Learning
- Other Applications and Summary

Wildlife Conservation

Human Behavior in Games

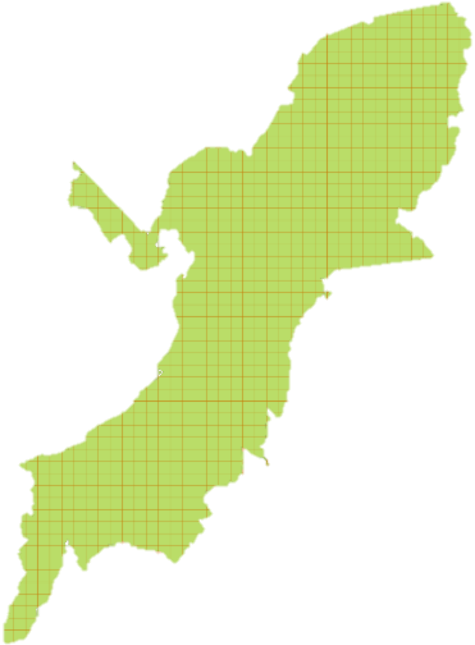
- Not always perfectly rational or behave as expected!
- Task: Predict where the poachers place snares



Learn from Human Subject Experiments

"A Game of Thrones": When Human Behavior Models Compete in Repeated Stackelberg Security Games. Debarun Kar, Fei Fang, Francesco Maria Delle Fave, Nicole Sintov, Milind Tambe. In AAMAS-15

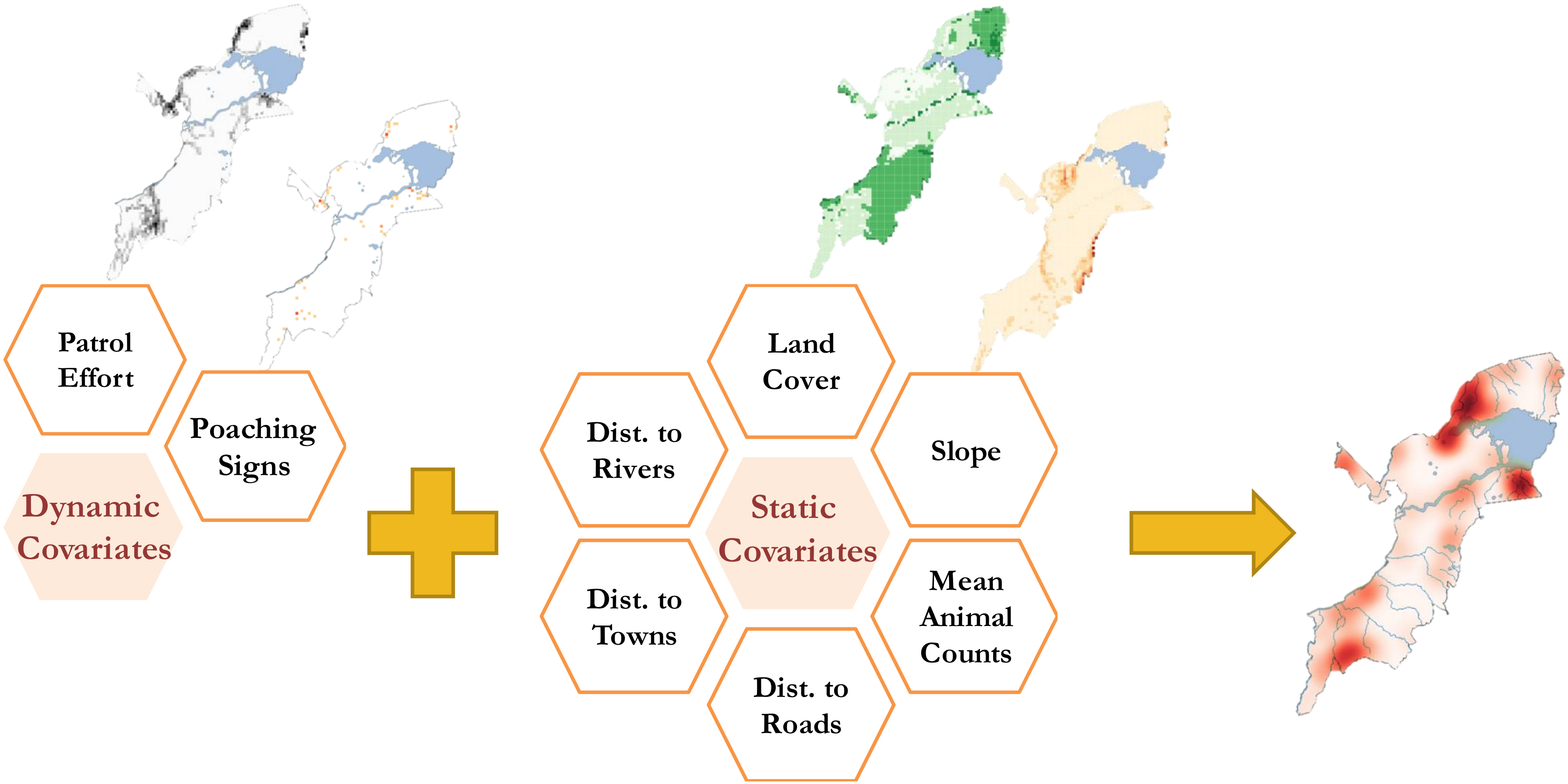
Learn from Real-World Data



- Raw Dataset for Queen Elizabeth National Park
  - Covers 2520 sq. km
  - Patrol and poaching recorded

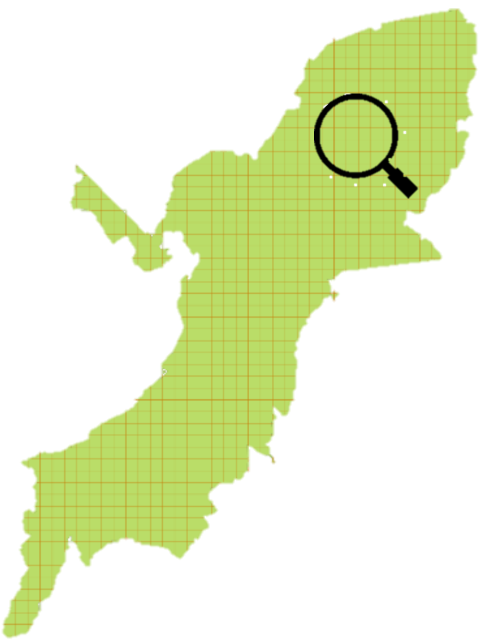
Collaborators: Wildlife Conservation Society, Uganda Wildlife Authority,
Rangers Pictures: Trip to Indonesia with World Wide Fund for Nature

Learn from Real-World Data



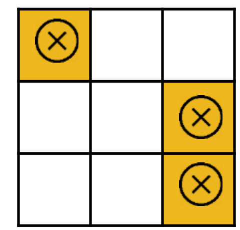
Each data point represent a 1km×1km area in a season

Challenge 1: Data Uncertainty

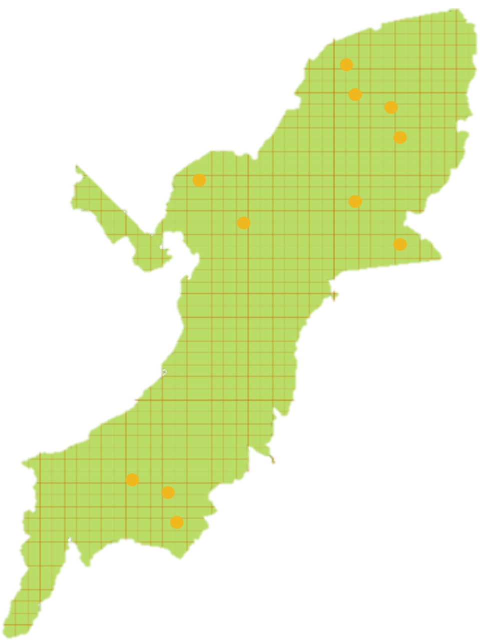


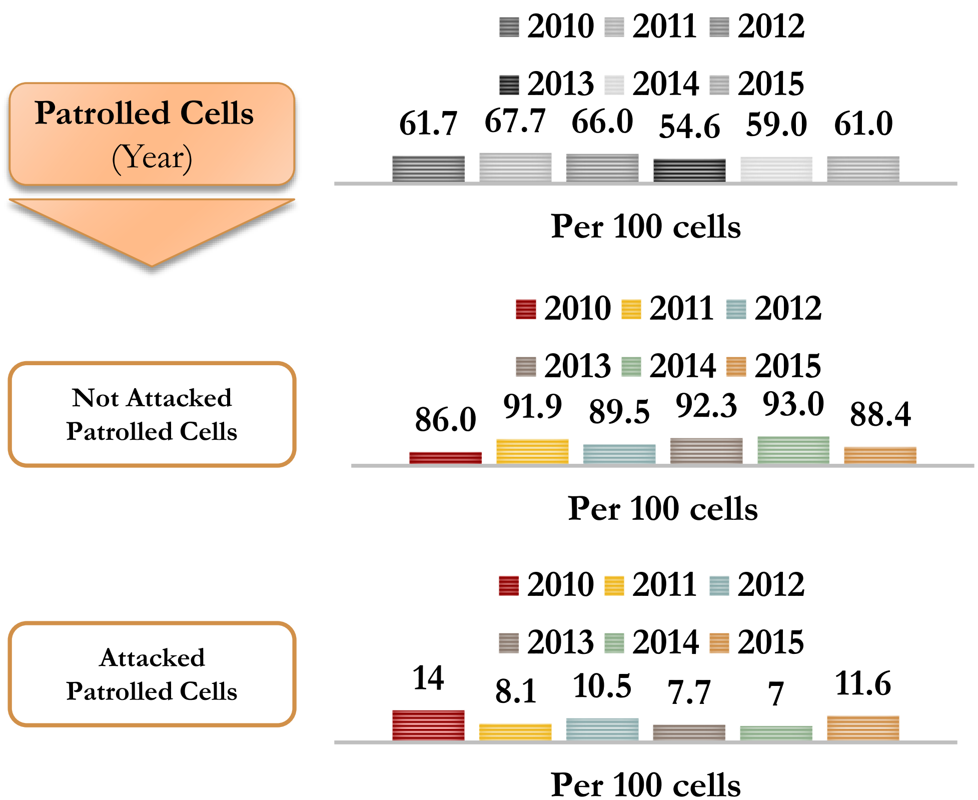
Attacked

Not Attacked



Challenge 2: Lack of Recorded Attacks





Quantal Response Model

- Classical model in behavioral game theory
- Probability of attacking target *j*

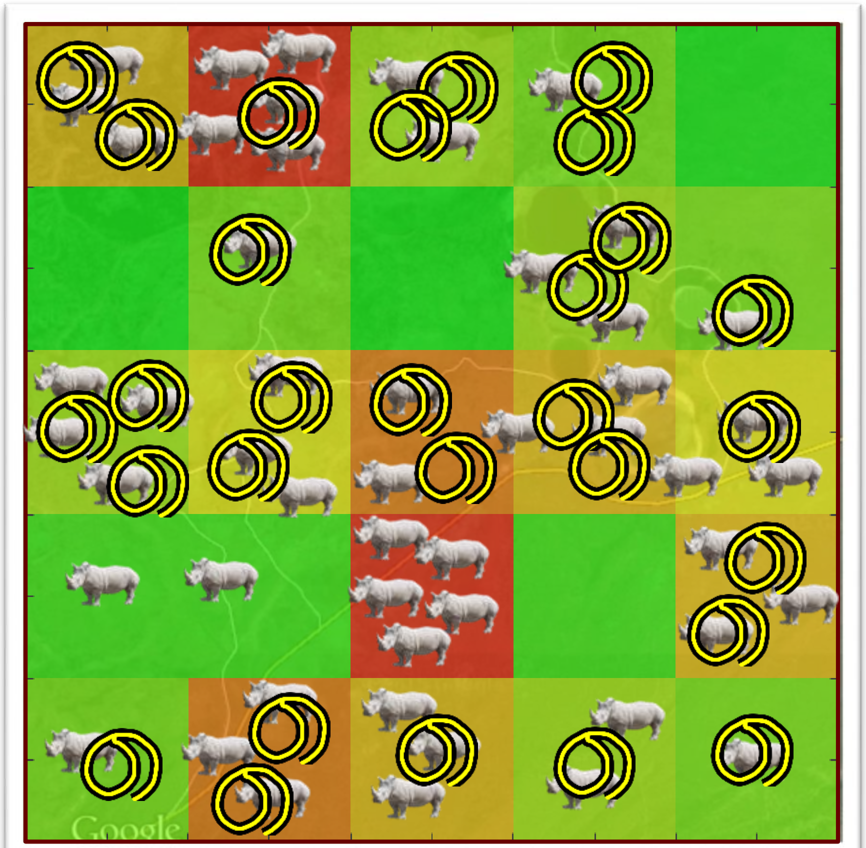
$$q_j = \frac{e^{\lambda * \mathrm{AttEU}_j(x)}}{\sum_i e^{\lambda * \mathrm{AttEU}_i(x)}}$$

- λ: represents error level (=0 means uniform random)
  - Maximal likelihood estimation (λ=0.76)
  - $\max_{\lambda} f(\lambda) = \sum_j N_j \log(q_j)$
  - Solved through gradient ascent $\lambda \leftarrow \lambda + \alpha \nabla_\lambda f(\lambda)$

McKelvey, R. D., & Palfrey, T. R. (1995). Quantal response equilibria for normal form games. Games and economic behavior, 10(1), 6-38.

Subjective Utility Quantal Response Model

- $\mathrm{SEU}_j = \sum_k w_k f_j^k, \ q_j = \frac{e^{\lambda * \mathrm{SEU}_j(x)}}{\sum_i e^{\lambda * \mathrm{SEU}_i(x)}}$



Past Success/Failure Induced Features +

Coverage Probability + Reward/Penalty

SUQR

Attack Probability

Nguyen, T. H., Yang, R., Azaria, A., Kraus, S., & Tambe, M. Analyzing the Effectiveness of Adversary Modeling in Security Games. In AAAI, 2013.

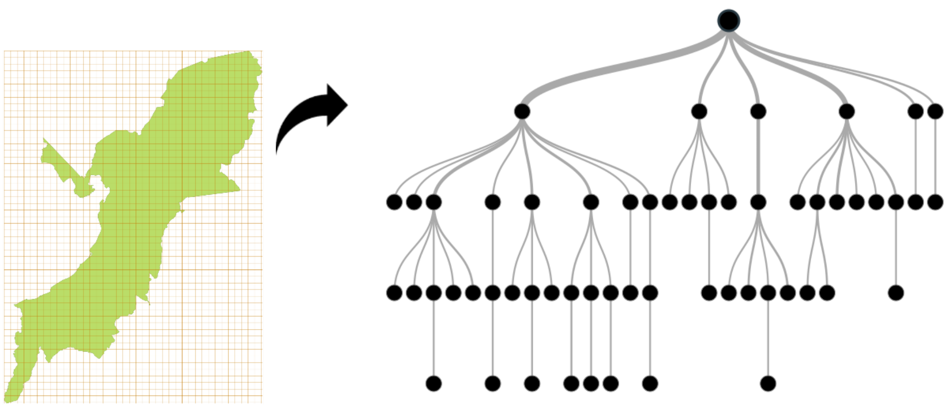
Adapted Behavioral Game Theory Models

- CAPTURE
  - Real-world Data
  - Dynamic Bayes Net: Time Dependency & Imperfect Observation

Thanh H. Nguyen, Arunesh Sinha, Shahrzad Gholami, Andrew Plumptre, Lucas Joppa, Milind Tambe, Margaret Driciru, Fred Wanyama, Aggrey Rwetsiba, Rob Critchlow, Colin Beale. CAPTURE: A New Predictive Anti-Poaching Tool for Wildlife Protection. In AAMAS, 2016.

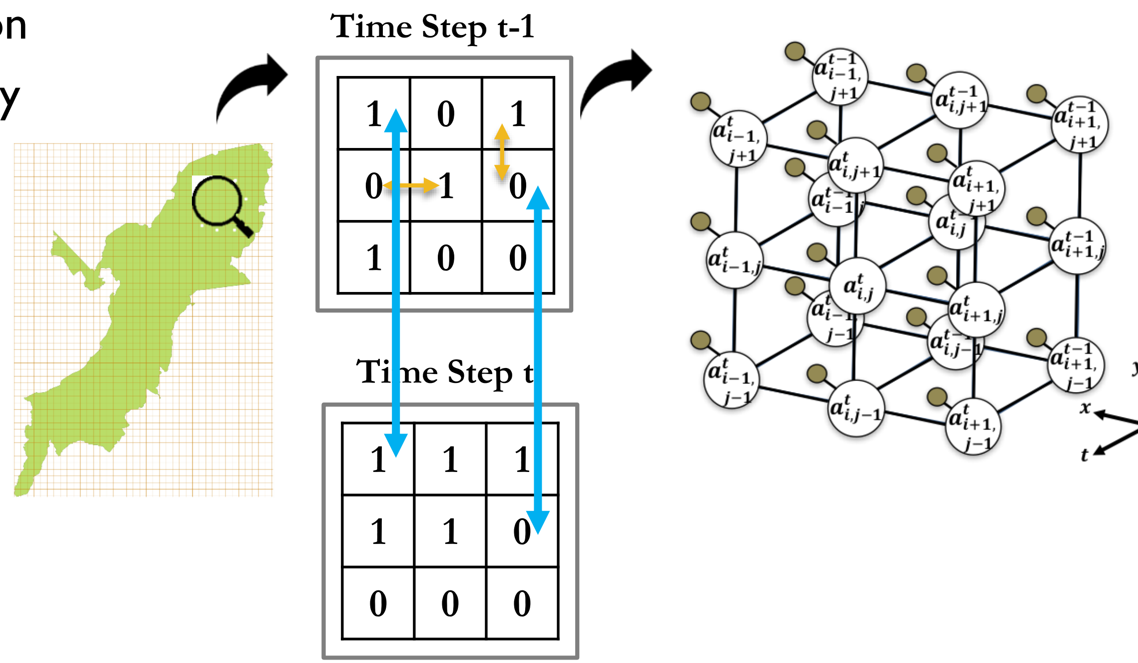
Decision Tree

- PROS
  - High speed
  - Learn global poachers behavior
  - Learn nonlinearity in geo-spatial predictor
- CONS
  - No explicit temporal dimension
  - No aspect for label uncertainty

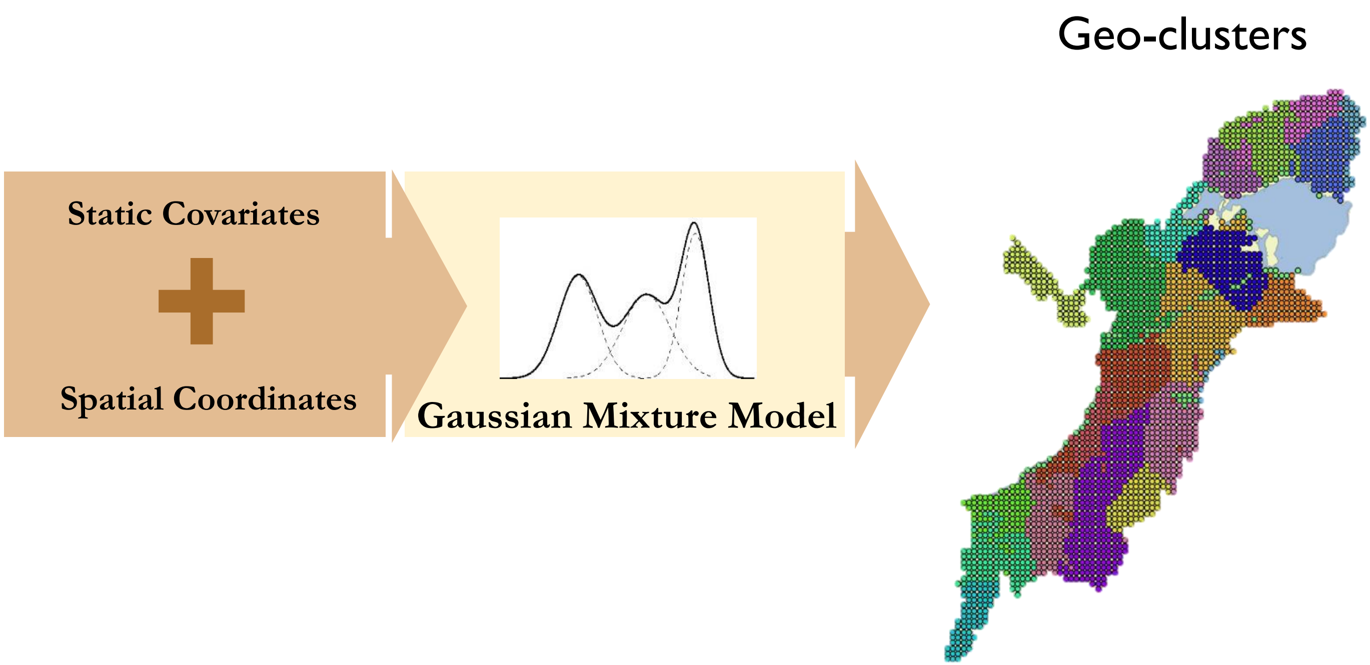


Markov Random Field

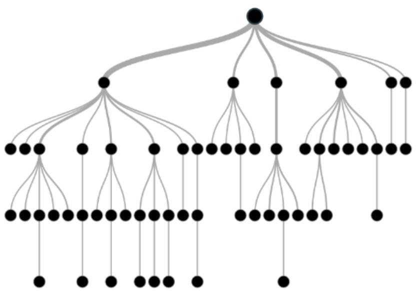
- PROS
  - Explicit spatial dimension
  - Explicit temporal dimension
  - Addresses label uncertainty
- CONS
  - Low speed
  - Data greedy

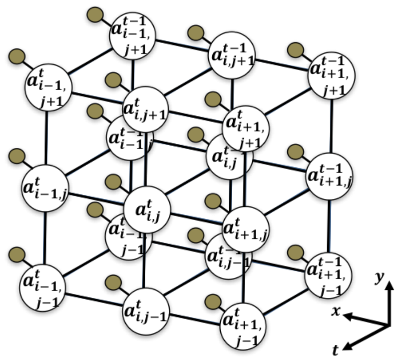


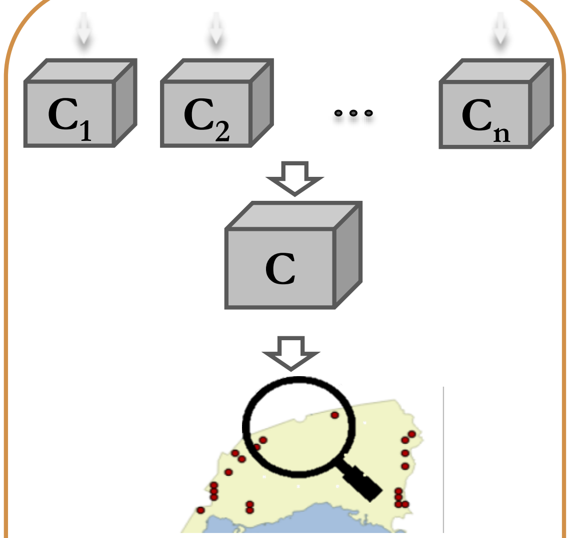
Hybrid Model



Hybrid Model

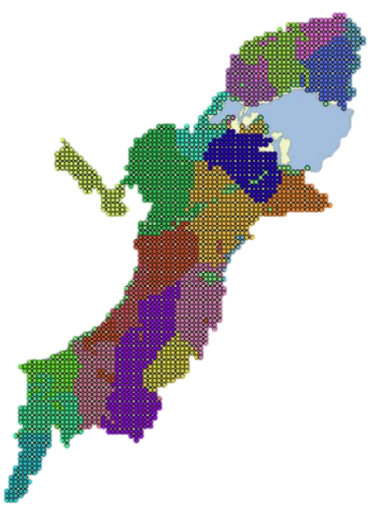




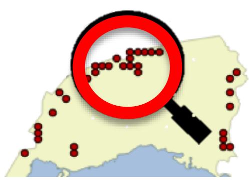


Bagging of Decision Trees

On Intensely Monitored Regions



Markov Random Fields



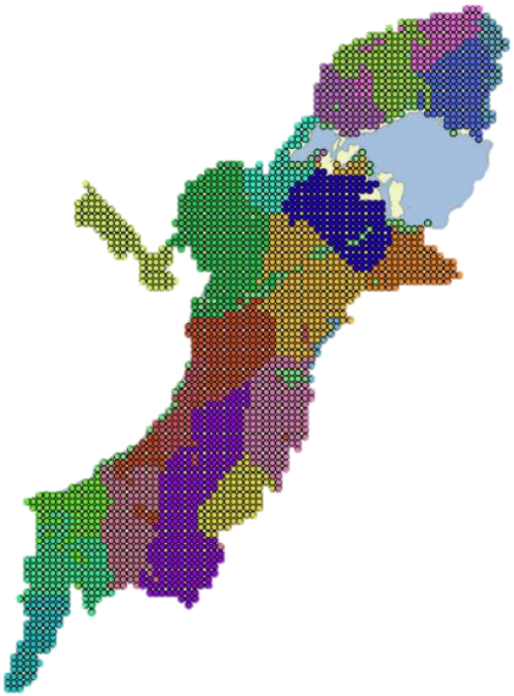
Decision Tree

+

Markov Random Fields

Taking it for a Test Drive: A Hybrid Spatio-temporal Model for Wildlife Poaching Prediction Evaluated through a Controlled Field Test. Shahrzad Gholami, Benjamin Ford, Fei Fang, Andrew Plumptre, Milind Tambe, Margaret Driciru, Fred Wanyama, Aggrey Rwetsiba, Mustapha Nsubaga, Joshua Mabonga. In ECML-PKDD 2017

Augment Dataset With Expert Knowledge



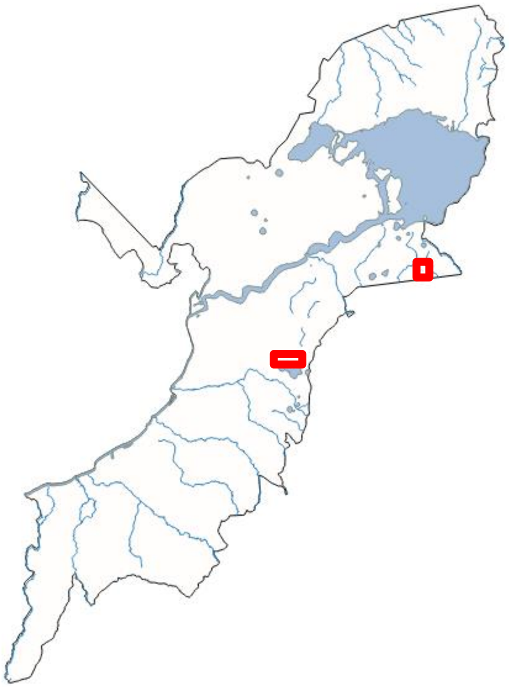
Cluster Index	Cluster Value in "predict_heatmap_i10.tif"	Estimated Snaring Threat (0~10)
1	1	7
2	2	3
3	3	8
4	4	7
5	5	3
6	6	2
7	7	8
8	8	3
9	9	0
10	10	0

- Negative sampling: sample from unpatrolled regions
- Positive sampling: Estimate from rangers' estimated scores
  - Collect answers for several sets of clusters C^1, C^2
  - Compute aggregated score a $s = \min\{s_1(C_i^1), s2(C_j^1), \ldots\}$, add unlabeled points as positive points if $s \geq 6$

Exploiting Data and Human Knowledge for Predicting Wildlife Poaching. Swaminathan Gurumurthy, Lantao Yu, Chenyan Zhang, Yongchao Jin, Weiping Li, Xiaodong Zhang, Fei Fang. In COMPASS-18

Field Test 1 in Uganda (1 month)

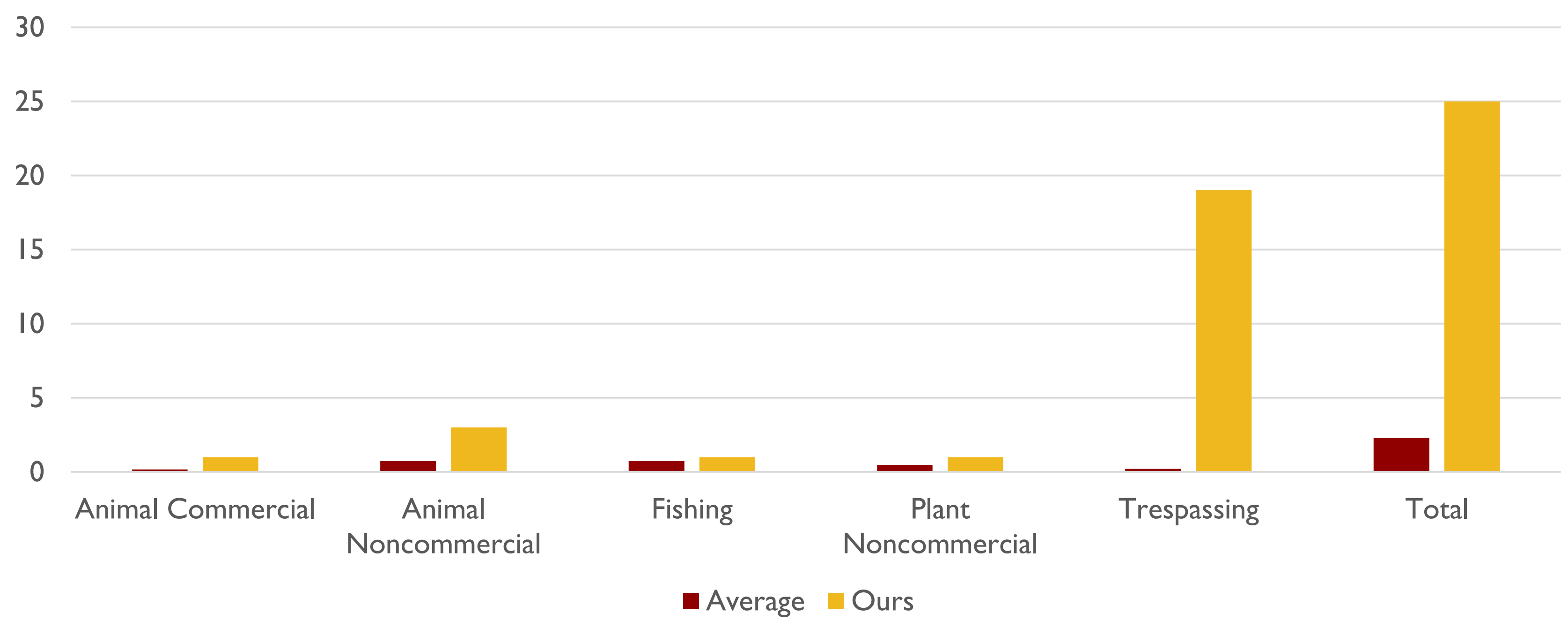
- Trespassing
  - 19 signs of litter, ashes, etc.
- Poached animals
  - 1 poached elephant
- Snaring
  - 1 active snare
  - 1 cache of 10 antelope snares
  - 1 roll of elephant snares
- Snaring hit rates
  - Outperform 91% of months



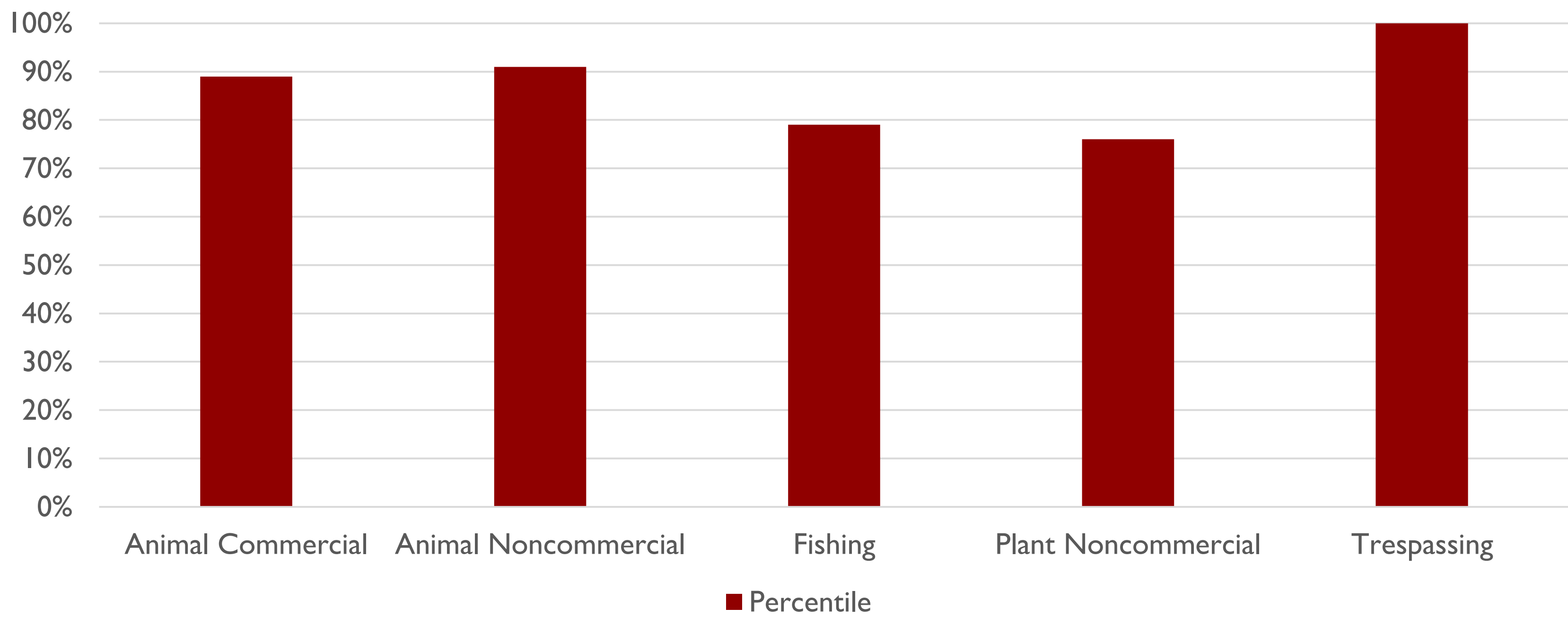
Historical Base Hit Rate	Our Hit Rate
Average: 0.73	3

Cloudy with a Chance of Poaching: Adversary Behavior Modeling and Forecasting with Real-World Poaching Data. Debarun Kar, Benjamin Ford, Shahrzad Gholami, Fei Fang, Andrew Plumptre, Milind Tambe, Margaret Driciru, Fred Wanyama, Aggrey Rwetsiba. In AAMAS-17

Field Test 1 in Uganda: Base rate comparison



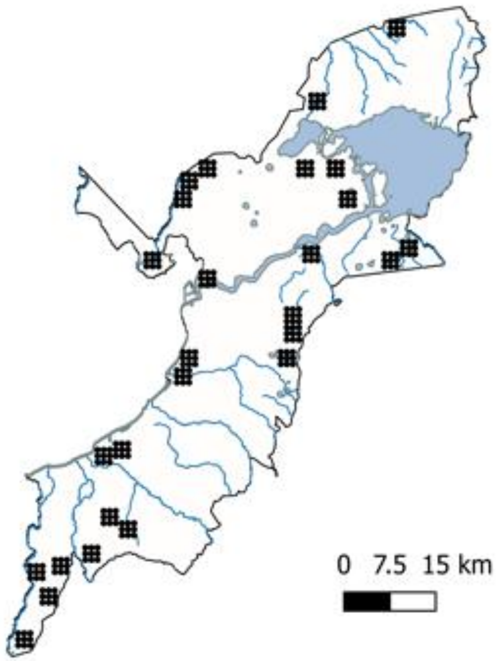
Field Test 1 in Uganda: % Months Exceeded Historical



Field Test 2 in Uganda (8 months)

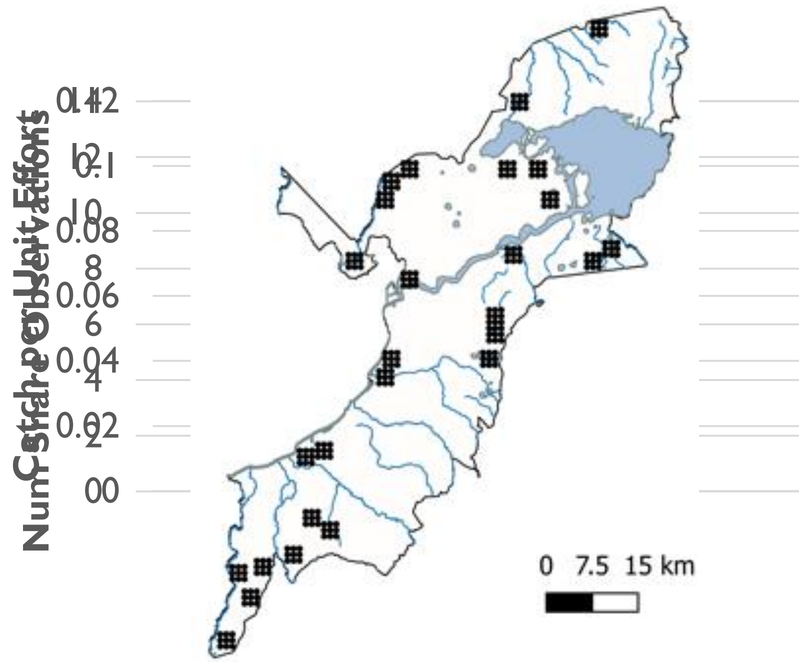
- 27 areas (9-sq km each)
- 454 km patrolled in total

- No point > 5 km from patrol post
- No area patrolled too much/rarely
- No overlapping areas
- <= 2 areas per patrol post



Field Test 2 in Uganda (8 months)

- 2 experiment groups
  - 1: >= 50% attack prediction rate
    - 5 areas
  - 2: < 50% attack prediction rate
    - 22 areas

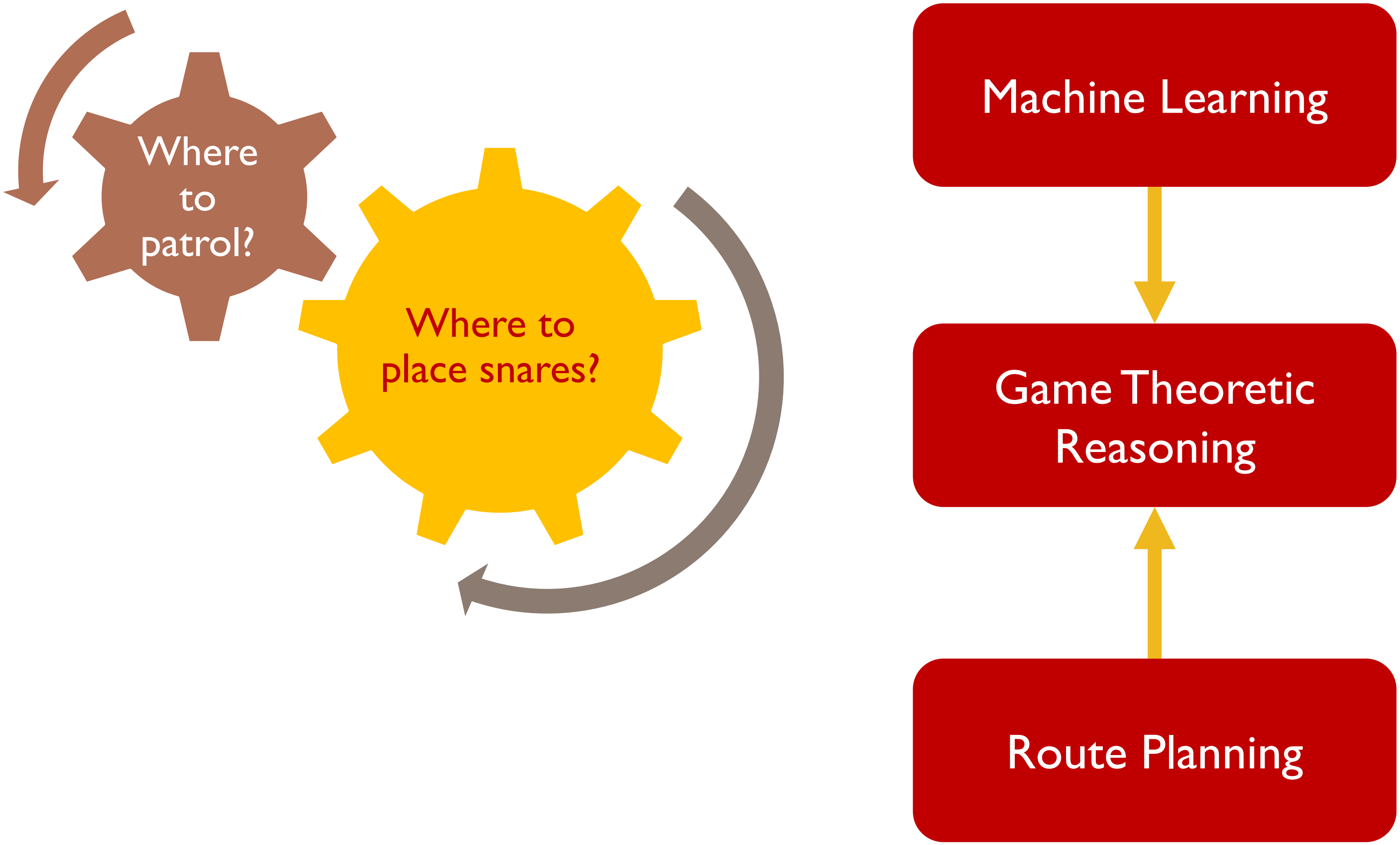


- Catch Per Unit Effort (CPUE)
  - Unit Effort = km walked

Field Test in China

- Two-day field test in October 2017: 22 snares
- 34 patrols from November 2017 to February 2018
  - 7 snares

From Prediction to Prescription



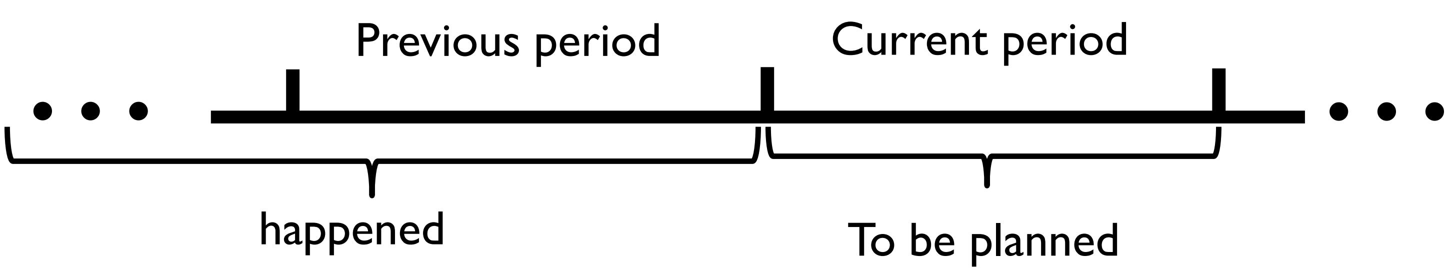
Game Theoretic Reasoning Based on Learned Model

- Find optimal patrol strategy given poachers respond to the patrol strategy according to learned model

- Challenges
  - Learned model is hard to represent using closed form function (e.g., decision tree)
  - Hard to scale up when considering scheduling constraints

Game Theoretic Reasoning Based on Learned Model

- Input: A machine learning model that predicts snares
- Output: an optimal patrolling strategy
- Goal: maximize catches of snares



Optimal Patrol Planning for Green Security Games with Black-Box Attackers. Haifeng Xu, Benjamin Ford, Fei Fang, Bistra Dilkina, Andrew Plumptre, Milind Tambe, Margaret Driciru, Fred Wanyama, Aggrey Rwetsiba, Mustapha Nsubaga, Joshua Mabonga. In GameSec-17: The 8th Conference on Decision and Game Theory for Security

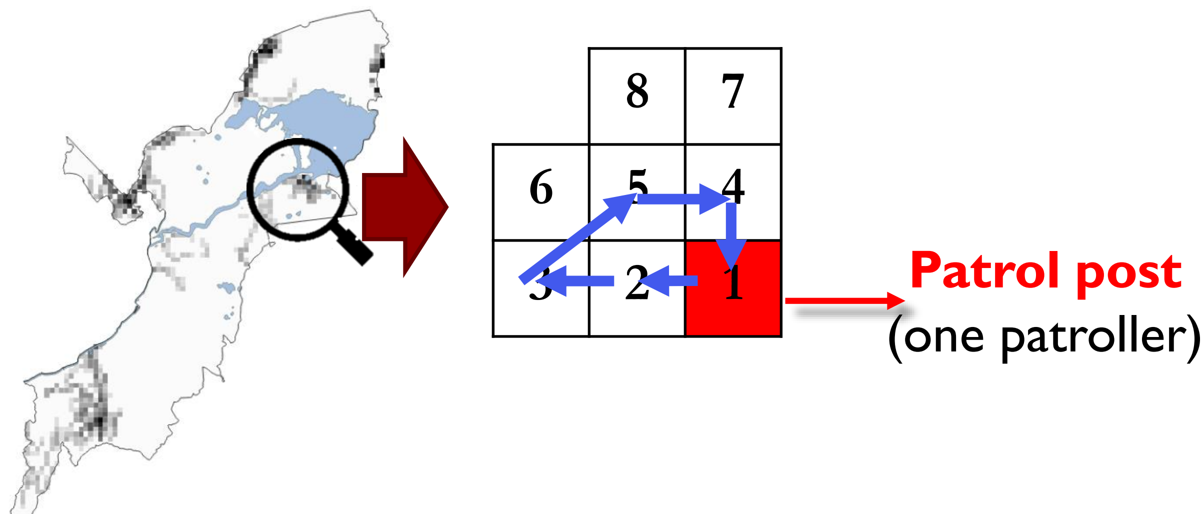
Game Theoretic Reasoning Based on Learned Model

For each cell i:

x_i: Current patrol effort at i → g_i → y_i: Prob. of detecting a snare at i in current period

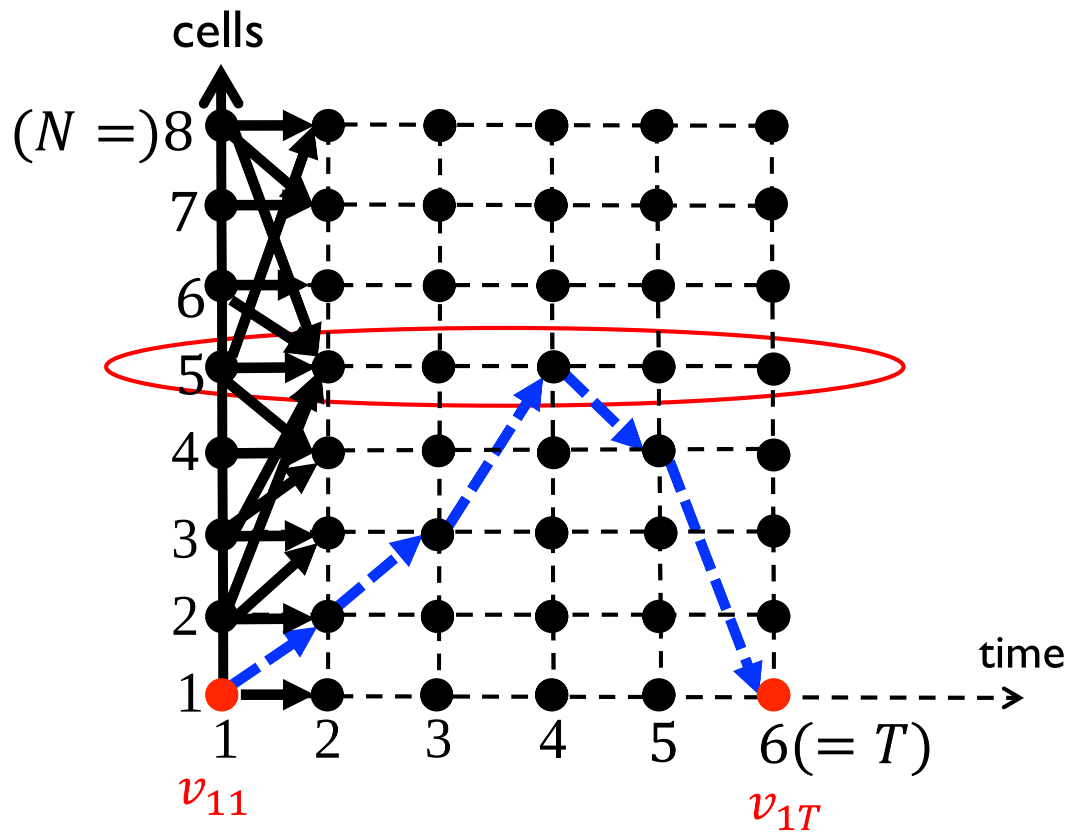
- Optimization problem: $\max_{x_i} \sum_i g_i(x_i)$

- However…



Game Theoretic Reasoning

- Observe: a pure strategy = a path from v_{11} to v_{1T}
- Claim: a mixed strategy $\Leftrightarrow$ one-unit fractional flow from v_{11} to v_{1T}
- Patrol effort at cell i = the aggregated flow through cell i
- Build a mixed integer linear program



Time-unrolled Graph

Game Theoretic Reasoning Based on Learned Model

- A MILP formulation

$$\text{maximize } \sum_{i=1}^{N} \left(g_i(0) + \sum_{j=1}^{m} z_i^j \cdot [g_i(j) - g_i(j-1)] \right)$$

$$\approx \max_{x_i} \sum_i g_i(x_i)$$

$$\begin{aligned}
\text{subject to } & x_i \geq \sum_{j=1}^{m} z_i^j \cdot [\alpha_j - \alpha_{j-1}], \\
& x_i \leq \alpha_1 + \sum_{j=1}^{m} z_i^j \cdot [\alpha_{j+1} - \alpha_j], \\
& z_i^1 \geq z_i^2 \ldots \geq z_i^m, \\
& z_i^j \in \{0, 1\},
\end{aligned}$$

$$x_i = z_i^1 + z_i^2 + \cdots$$

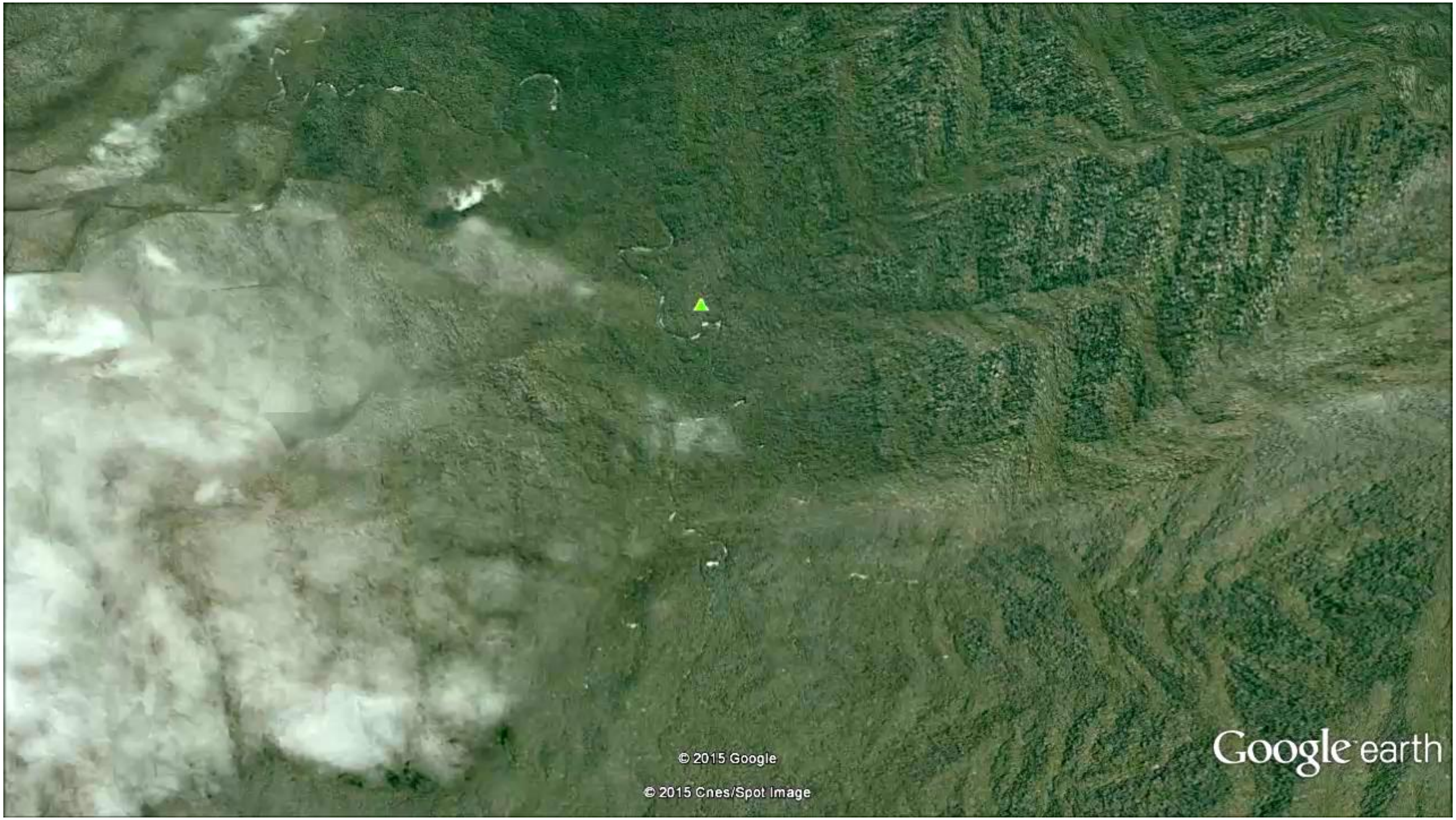
$$x_i = \sum_{t=1}^{T} \left[\sum_{e \in \sigma^+(v_{t,i})} f(e) \right],$$

Patrol effort at cell i = the aggregated flow through cell i

$$\begin{aligned}
& \sum_{e \in \sigma^+(v_{t,i})} f(e) = \sum_{e \in \sigma^-(v_{t,i})} f(e), \\
& \sum_{e \in \sigma^+(v_{T,1})} f(e) = \sum_{e \in \sigma^-(v_{1,1})} f(e) = 1 \\
& 0 \leq x_i \leq 1, \qquad 0 \leq f(e) \leq 1,
\end{aligned}$$

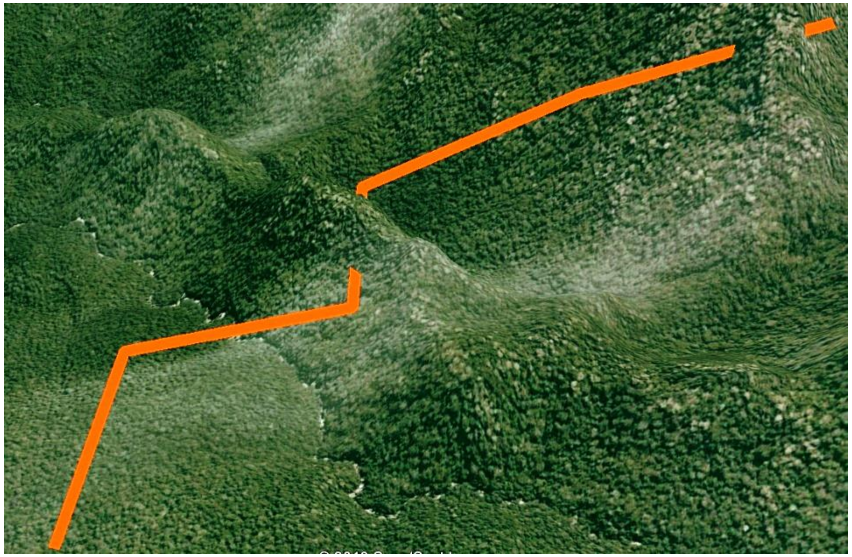
f is a unit flow

Complex Terrain

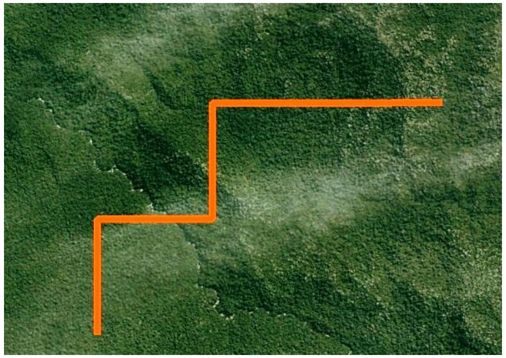


Complex Terrain

Patrol Route (3D)

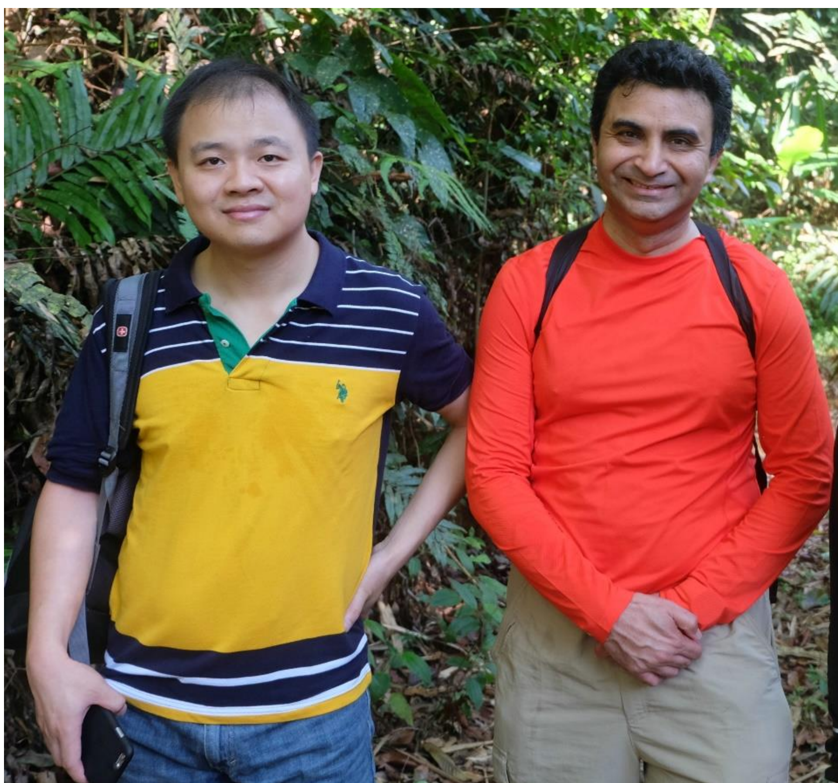


Patrol Route (2D)

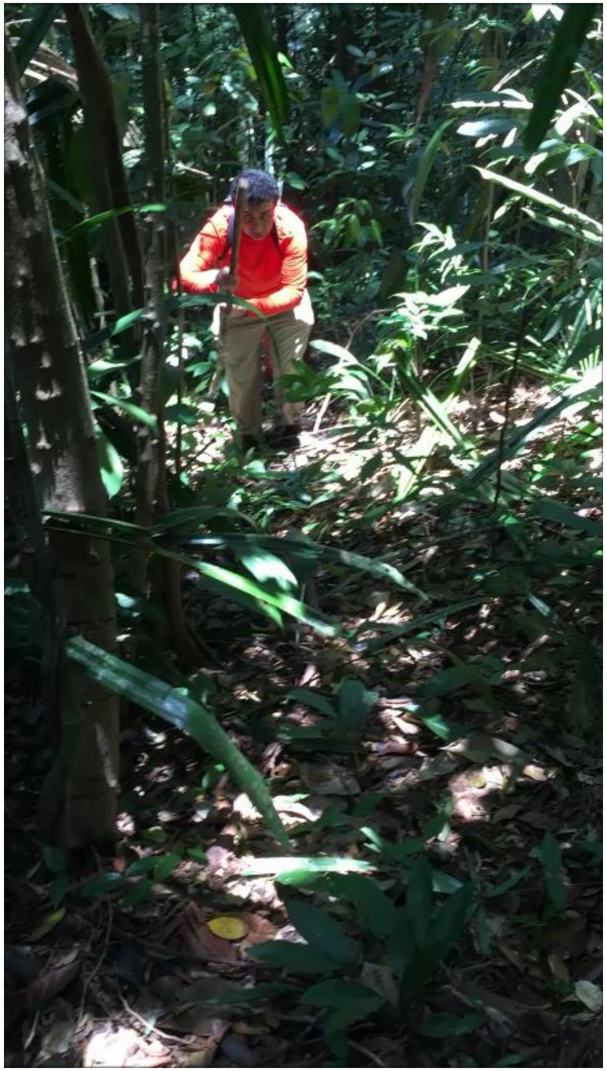


Trial Patrol in the Field

- 8-hour patrol in April 2015: patrolling is not easy!

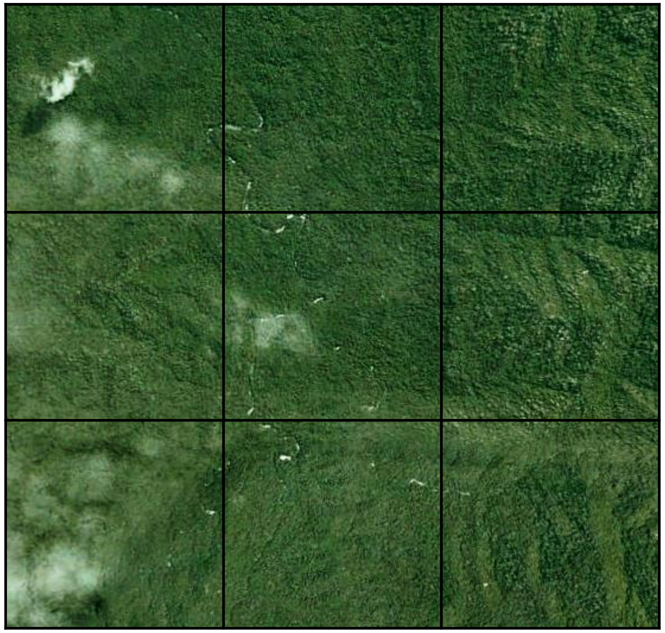


Spatial Constraint



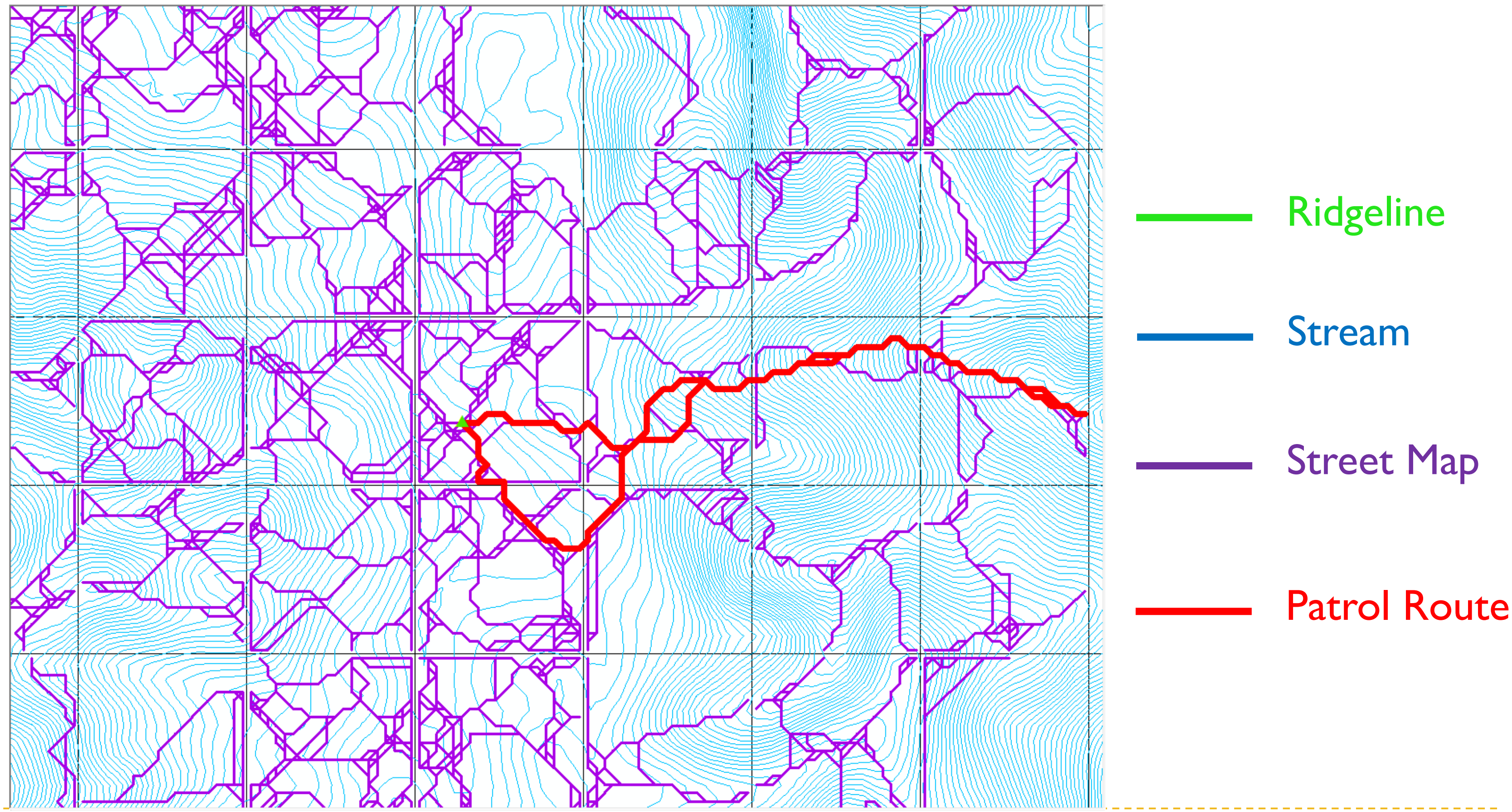
Spatial Constraint

- Grid based → Route based
- Hierarchical modeling: Focus on terrain features
- Build virtual street map

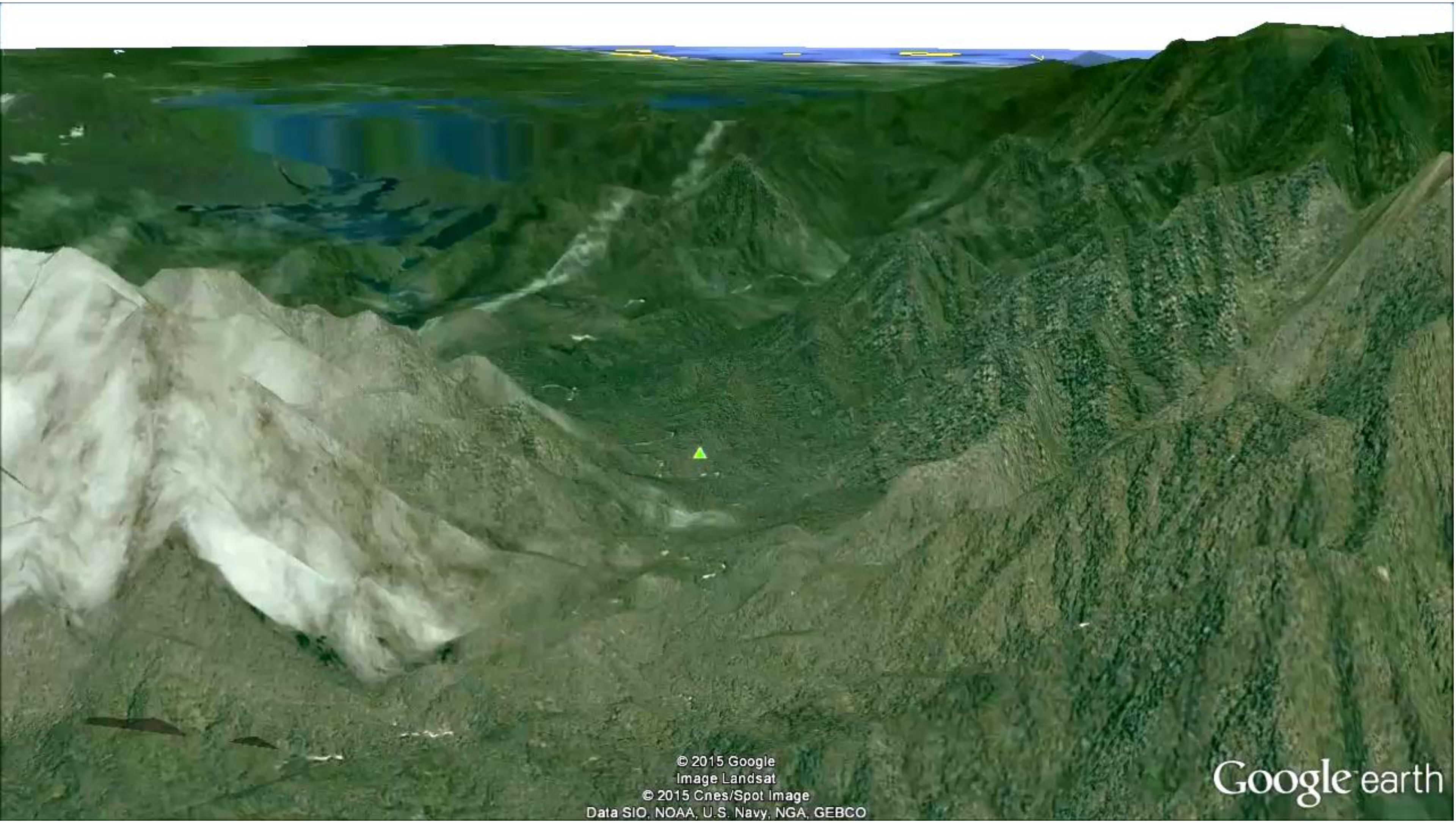


Spatial Constraint

- Hierarchical model: Focus on terrain feature



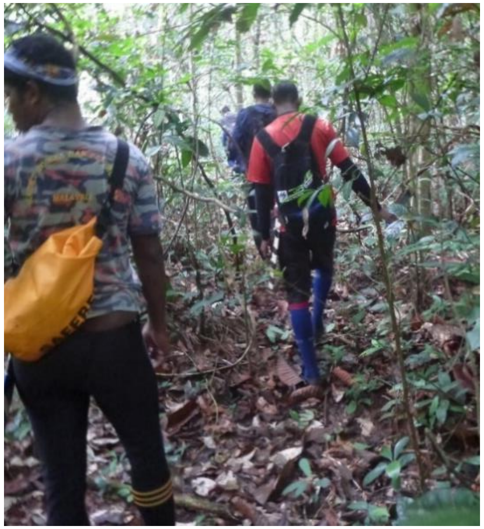
Patrol Route Design



Deploying PAWS: Field Optimization of the Protection Assistant for Wildlife Security. Fei Fang, Thanh H. Nguyen, Rob Pickles, Wai Y. Lam, Gopalasamy R. Clements, Bo An, Amandeep Singh, Milind Tambe, Andrew Lemieux. In IAAI-16

Field Test in Malaysia

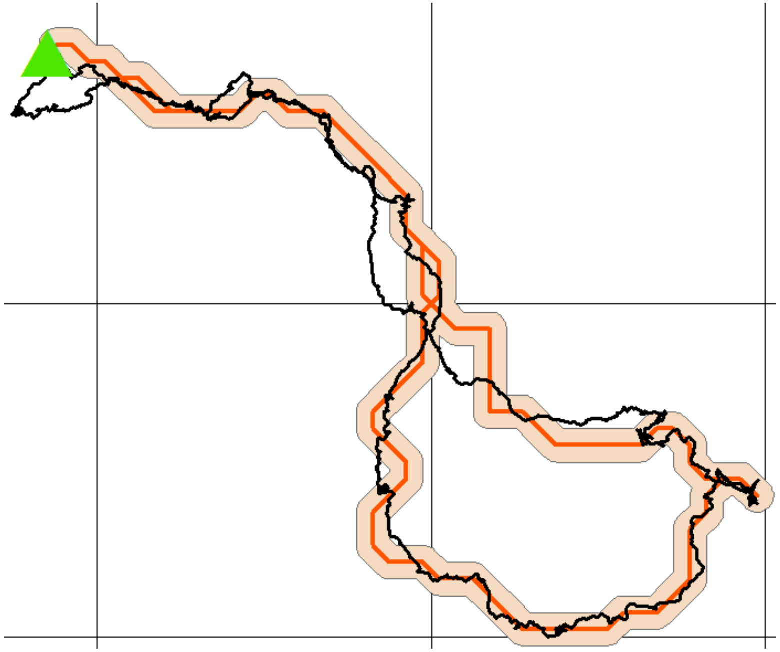
- In collaboration with Panthera, Rimba
- Regular deployment since July 2015 (Malaysia)



Real-World Deployment

Grid Based

Route Based



Real-World Deployment

Animal Footprint

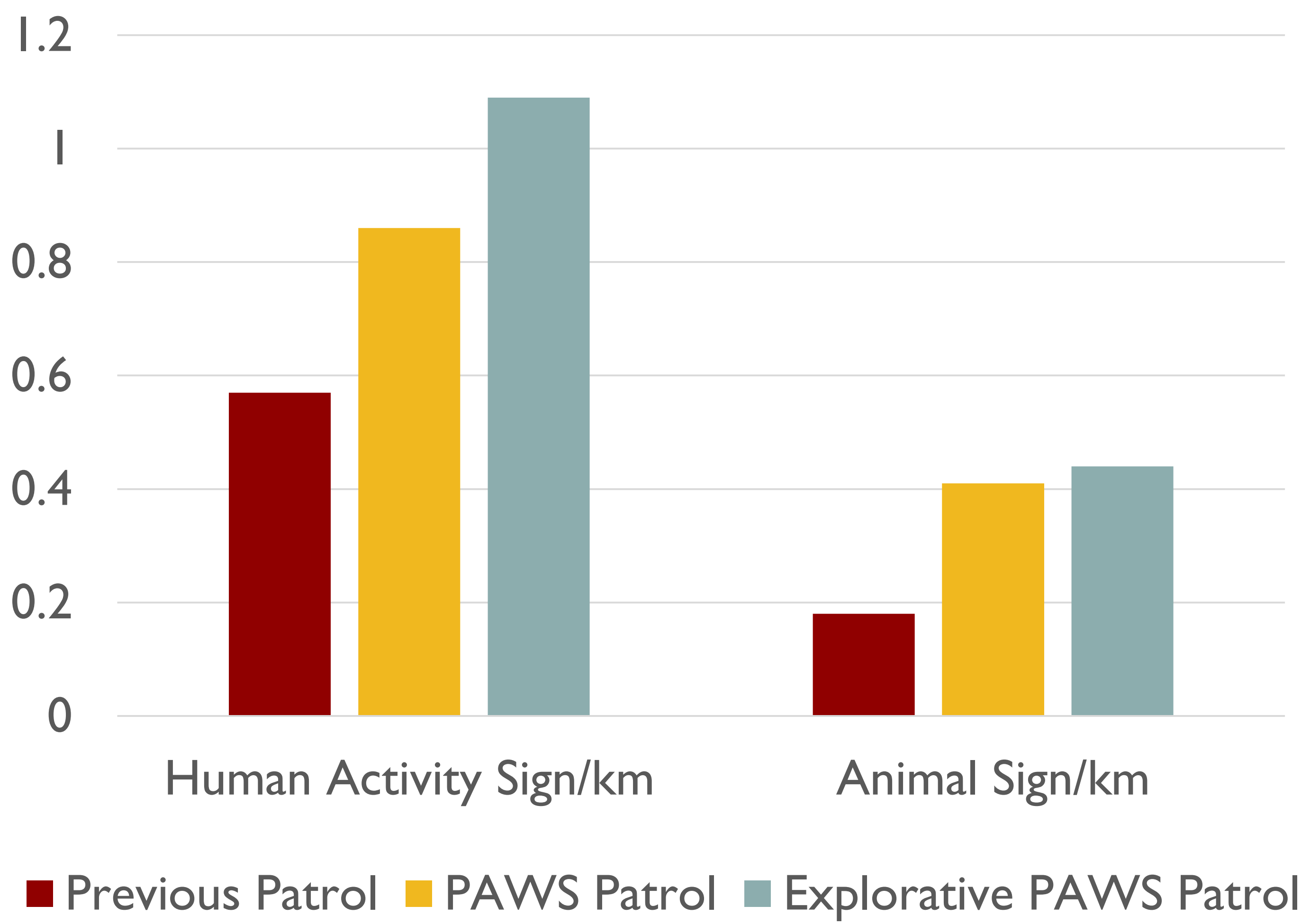
Tiger Sign

Tree Mark

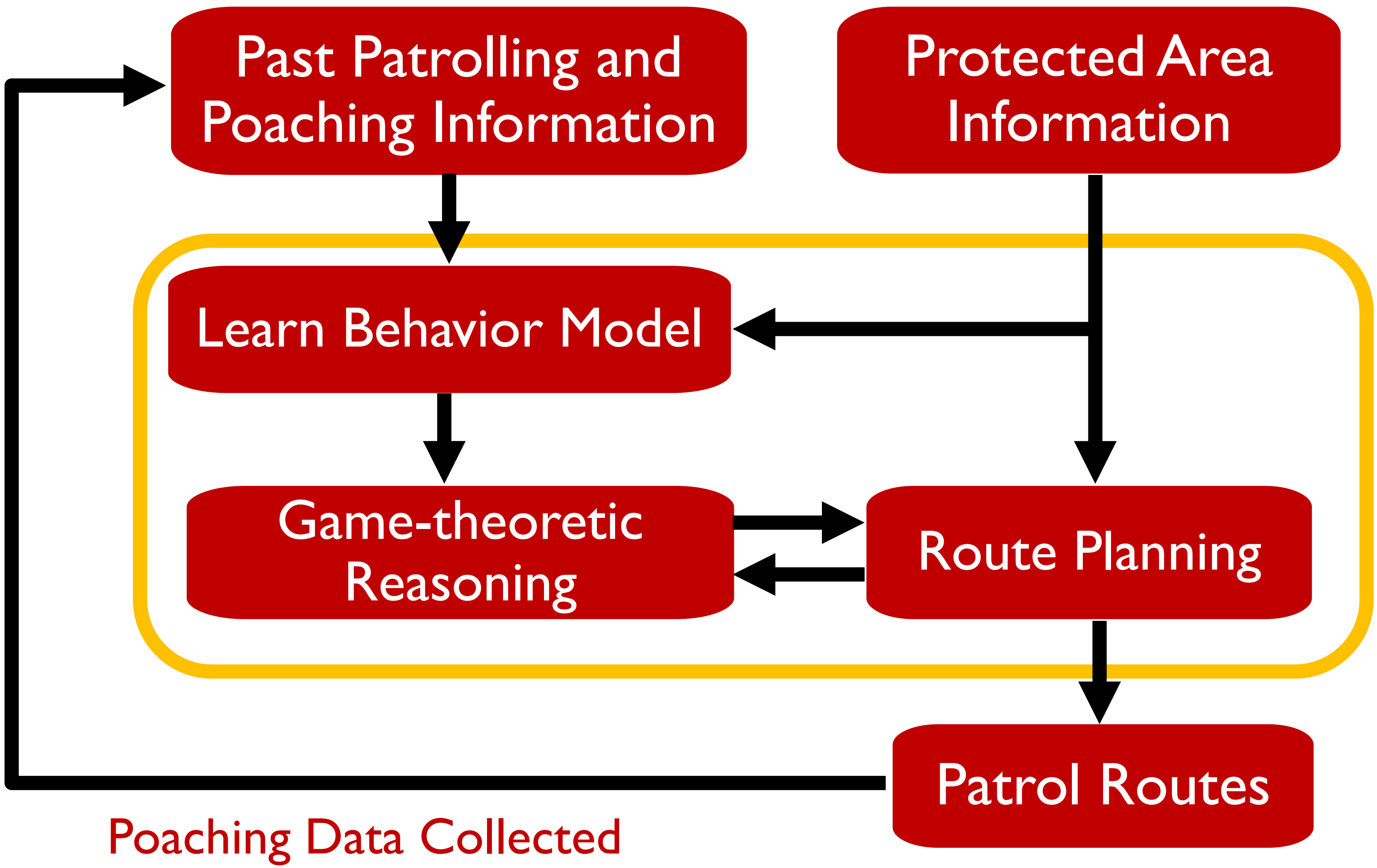
Camping Sign

Lighter

Real-World Deployment

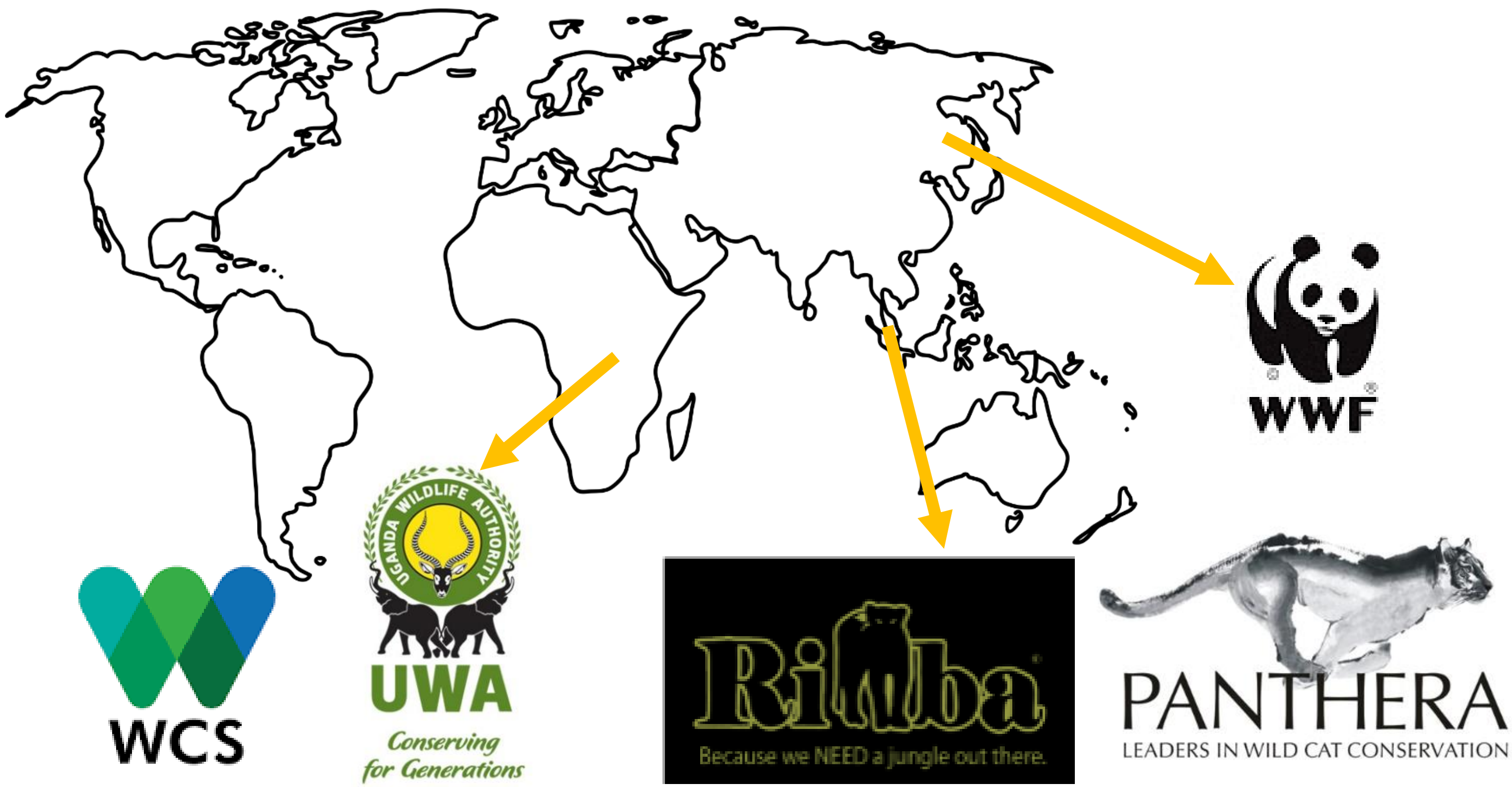


PAWS: Protection Assistant for Wildlife Security



PAWS: Protection Assistant for Wildlife Security

- PAWS is deployed in the field
  - Saved animals!



Outline

- Games with Human Players for Real-world Applications
  - Wildlife Conservation
- End-to-End Learning and Decision Making in Games
  - A differentiable learning framework for learning game parameters
- Learning-Powered Strategy Computation in Large Games
  - Leveraging Deep Reinforcement Learning
- Other Applications and Summary

What game are we/they playing?

- Common criticism: game parameters are fully known
  - E.g. target importance
- How to learn parameters of **2-player zero sum games** from opponents' or players' actions?

What game are we playing? End-to-end learning in normal and extensive form games.
Chun Kai Ling, Fei Fang, J. Zico Kolter. In IJCAI-ECAI-18

Forward Problem: Game Solving

			
	0	-1	1
	1	0	-1
	-1	1	0

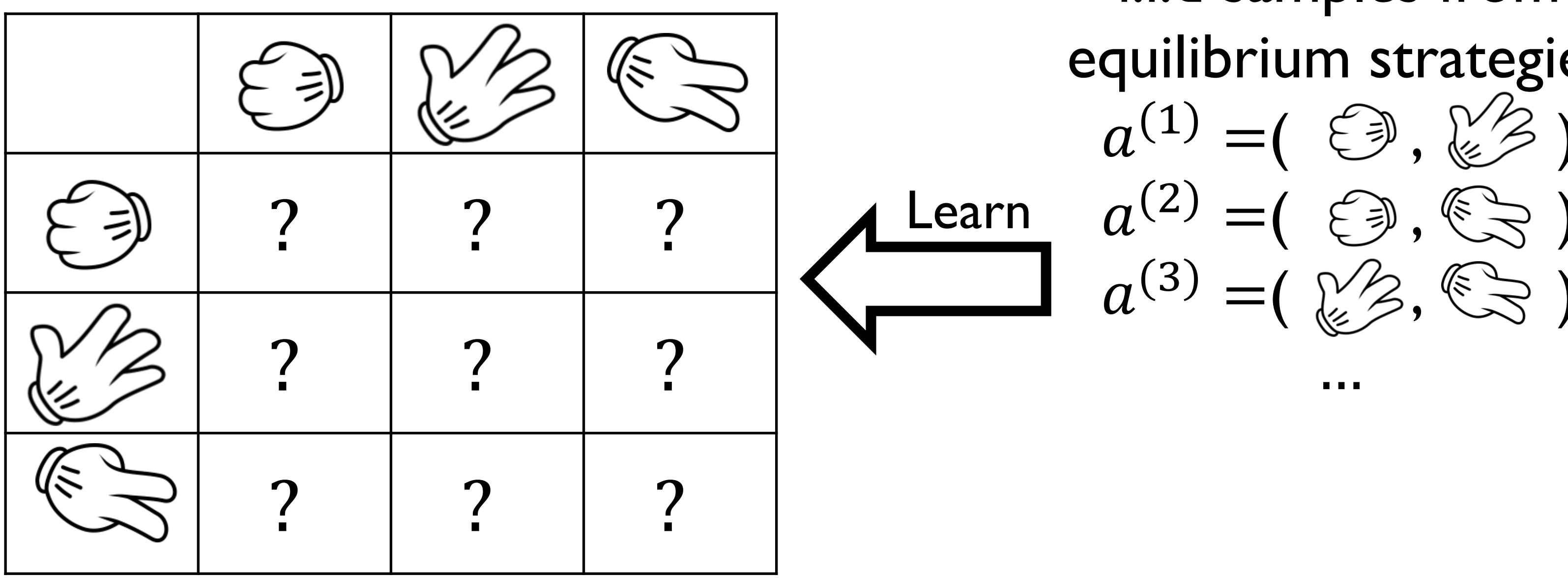
Solve

Equilibrium strategies

$$u^* = v^* = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$$

Inverse Problem: Game Learning

i.i.d samples from equilibrium strategies



What game are we/they playing?

- Previous work on this topic
  - Directly learn good strategies from data (e.g. Letchford et al. , 2009)
  - Rely on special game structures (Vorobeychik et al. , 2007)
  - Computational Rationalization framework (Waugh et al. , 2011)

What game are we playing? End-to-end learning in normal and extensive form games.
Chun Kai Ling, Fei Fang, J. Zico Kolter. In IJCAI-ECAI-18

Differentiable Learning

			
	0	$-b_1$	$-b_2$
	b_1	0	$-b_3$
	$-b_1$	b_3	0

$\leftarrow$ Learn

i.i.d samples from equilibrium strategies

$a^{(1)}$ =( , )

$a^{(2)}$ =( , )

$a^{(3)}$ =( , )

- Guess the value of b_i
- Compute equilibrium of guessed game
- Check if the computed equilibrium consistent with data
- Adjust the value of b_i to increase consistency
- Repeat until satisfied

→ Update $b_i := b_i - \frac{\partial L}{\partial b_i}$

NE and QRE in Zero-Sum Games

Recall LP for computing NE

$$\min_{u,x} x$$

s.t. $x \geq \sum_i u_i P_{ij}, \forall j$

$\sum_i u_i = 1, u_i \geq 0, \forall i$

Recall Quantal Response

$$q_j = \frac{e^{\lambda * \mathrm{AttEU}_j(x)}}{\sum_i e^{\lambda * \mathrm{AttEU}_i(x)}}$$

Nash Equilibrium

- Assumes perfect rationality
- May have multiple equilibria
- Discontinuous w.r.t. P

$$\min_u \max_v u^T P v$$

s.t.

$$1^T u = 1, u \geq 0$$
$$1^T v = 1, v \geq 0$$

Quantal Response Equilibrium

- Captures bounded rationality
- Unique
- Continuous w.r.t. P

$$\min_u \max_v u^T P v - \sum_i v_i \log v_i + \sum_i u_i \log u_i$$

s.t.

$$1^T u = 1, u \geq 0$$
$$1^T v = 1, v \geq 0$$

$$u_i^* = \frac{\exp(Pv)_i}{\sum_q \exp(Pv)_q}, v_j^* = \frac{\exp(P^T u)_j}{\sum_q \exp(P^T u)_q}$$

Learning of normal form games

- QRE = solution of min-max convex-concave problem

$$\min_u \max_v u^T P v - \sum_i v_i \log v_i + \sum_i u_i \log u_i$$
$$1^T u = 1,\ 1^T v = 1$$

- KKT conditions:

$$Pv + \log(u) + 1 + \mu 1 = 0$$
$$P^T u - \log(v) - 1 + \nu 1 = 0$$
$$1^T u = 1,\ 1^T v = 1$$

Recall: Newton's Method for 1-D:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Generally, for nonlinear system

$$J_F(x_n)(x_{n+1} - x_n) = -F(x_n)$$

- Forward pass: Apply Newton's Method

$$\begin{bmatrix} diag(\frac{1}{u}) & P & & \\ & & 1 & 0 \\ P^T & -diag\left(\frac{1}{v}\right) & 0 & 1 \\ & & 0 & 0 \\ 1^T & 0 & 0 & 0 \\ 0 & 1^T & & \end{bmatrix} \begin{bmatrix} \Delta u \\ \Delta v \\ \Delta \mu \\ \Delta \nu \end{bmatrix} = - \begin{bmatrix} Pv + \log(u) + 1 + \mu 1 \\ P^T u - \log(v) - 1 + \nu 1 \\ 1^T u - 1 \\ 1^T v - 1 \end{bmatrix}$$

Learning of normal form games

- Backward pass: Gradients of P may be obtained via the implicit function theorem

$$\nabla_P L = y_u v^T + u y_v^T,$$

where

$$\begin{bmatrix} y_u \\ y_v \\ y_\mu \\ y_\nu \end{bmatrix} = \begin{bmatrix} \mathrm{diag}(\frac{1}{u}) & P & 1 & 0 \\ P^T & -\mathrm{diag}(\frac{1}{v}) & 0 & 1 \\ 1^T & 0 & 0 & 0 \\ 0 & 1^T & 0 & 0 \end{bmatrix}^{-1} \begin{bmatrix} -\nabla_u L \\ -\nabla_v L \\ 0 \\ 0 \end{bmatrix}$$

Learning in the presence of features

			
	0	$-b_1(x)$	$b_2(x)$
	$b_1(x)$	0	$-b_3(x)$
	$-b_2(x)$	$b_3(x)$	0

Learn

i.i.d samples from equilibrium strategies

$a^{(1)}$ =( , )

$a^{(2)}$ =( , )

$a^{(3)}$ =( , )

...

Context

$$x^{(1)} = [0.1, 0.5]$$

$$x^{(2)} = [0.3, 0.7]$$

$$\dots$$

Learning in the presence of features

- Figure out which features attract/discourage attackers
  - Better understand attacker's interests
  - Design better configurations which favor defenders

- Predict each player's mixed strategy given an *new* environment
  - In practice, environment is changing over time

Learning in the presence of features

- *Context* (feature) $x^{(i)}$ and payoff matrix $P_\Phi(x^{(i)})$, parameterized by Φ

- Each player acts according to a mixed strategy (u, v) given by the QRE of $P_\Phi(x^{(i)})$, giving realizations $a^{(i)}$

- Objective: Learn Φ from $\{x^{(i)}, a^{(i)}\}$

End-to-end learning

Algorithm 1: Learning parameters Φ using SGD

Input: training data $\{(x^{(i)}, a^{(i)})\}$, learning rate η, Φ_{init}

for *ep in* $\{0, \ldots, ep_{max}\}$ **do**

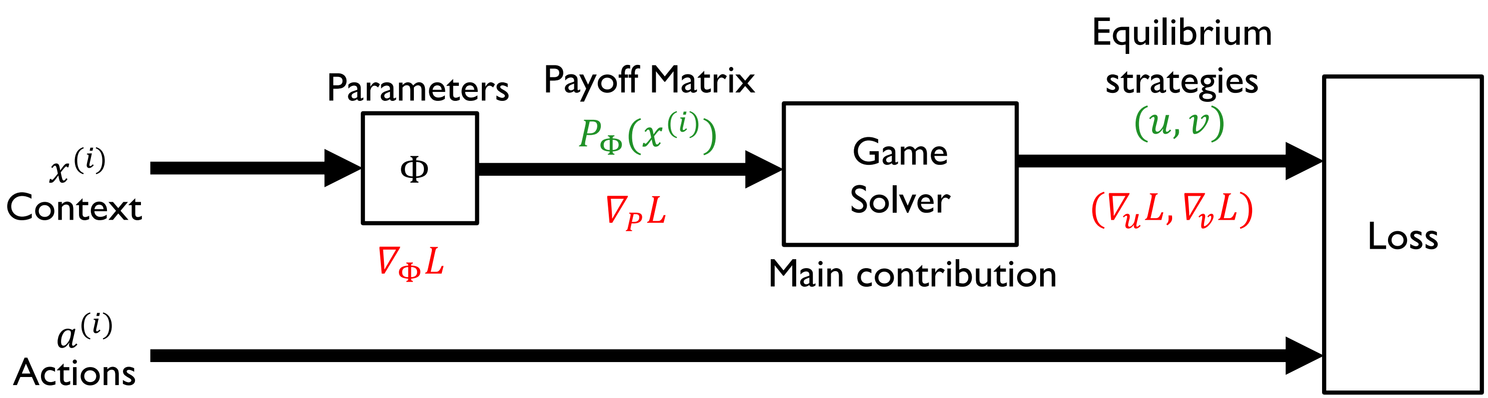
  Sample $(x^{(i)}, a^{(i)})$ from training data;

  Forward pass: Compute $P_\Phi(x^{(i)})$, QRE (u, v) and loss $L(a^{(i)}, u, v)$;

  Backward pass: Compute gradients $\nabla_u L, \nabla_v L, \nabla_P L, \nabla_\Phi L$;

  Update parameters: $\Phi \leftarrow \Phi - \eta \nabla_\Phi L$;

end



Extensive form Games

- Let (u, v) be strategies in sequence form
- Equilibrium is expressed as solution using *dilated entropy regularization (*Equivalent to solving QRE for the reduced normal form)

$$\min_u \max_v u^T P v - \sum_i \sum_a v_a \log(\frac{v_a}{v_{p_i}}) + \sum_i \sum_a u_a \log(\frac{u_a}{u_{p_i}})$$
$$Eu = e, Fv = f$$

$$\nabla_P L = y_u v^T + u y_v^T,$$

where

$$\begin{bmatrix} y_u \\ y_v \\ y_\mu \\ y_\nu \end{bmatrix} = \begin{bmatrix} -\Xi(u) & P & E^T & 0 \\ P^T & \Xi(v) & 0 & F^T \\ E & 0 & 0 & 0 \\ 0 & F & 0 & 0 \end{bmatrix}^{-1} \begin{bmatrix} -\nabla_u L \\ -\nabla_v L \\ 0 \\ 0 \end{bmatrix}$$

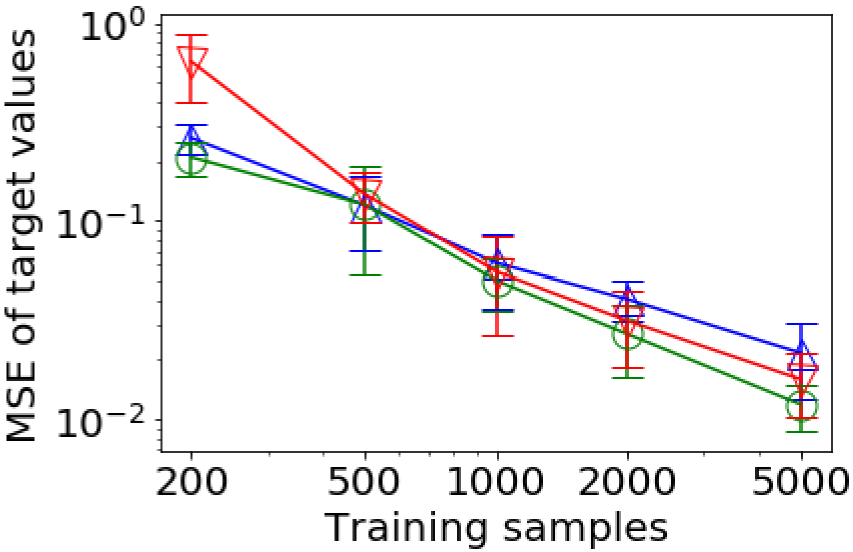
Resource Allocation Security Game

- Defender: r resources, n targets
  - Can allocate multiple resources to one target
- Attacker choose a target to attack
- Each target has value R_i
- If target i is protected by x resources and is attacked: $U_a = \frac{R_i}{2^x} = -U_d$
- Attacker may learn R_i from observed defender actions

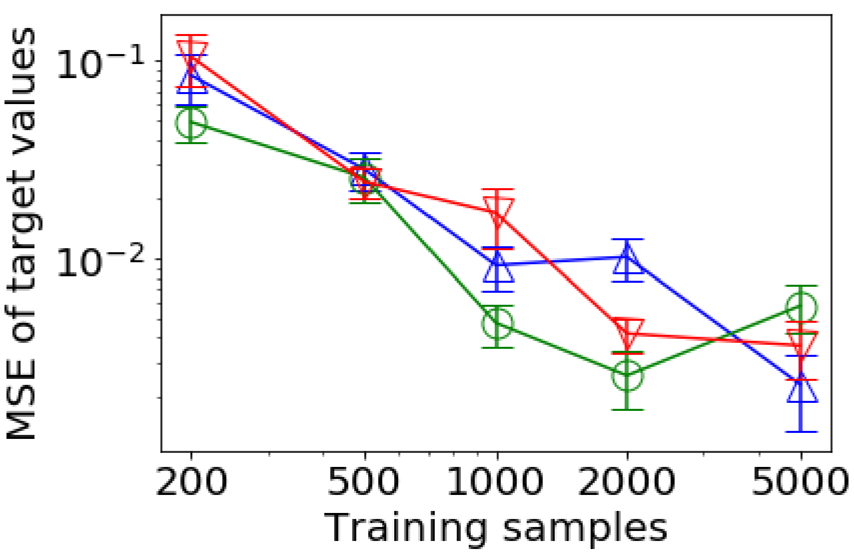
- Extend to T-stage game

Resource Allocation Security Game

- $n = 2, r = 5$



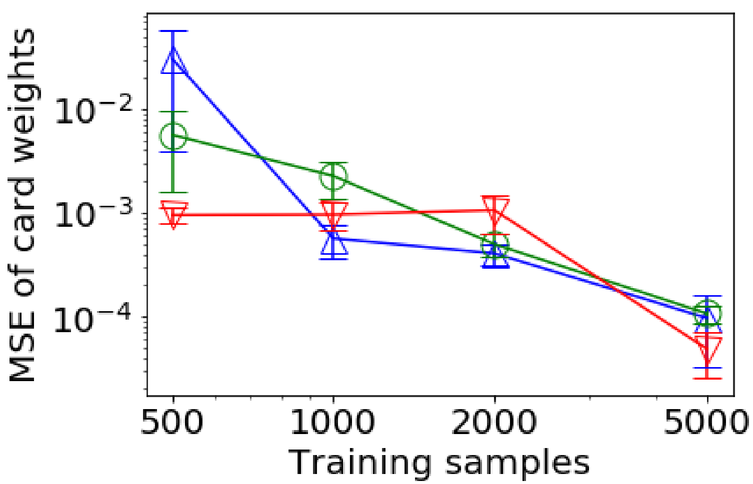
T= 1

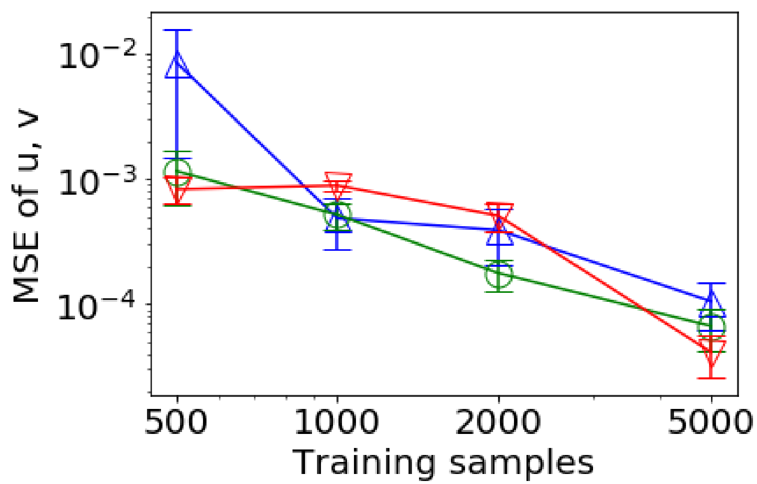


T= 2

One-Card Poker

- Learn players' belief of card distribution
- Variant of Kuhn Poker with 4 cards, with *non-uniform* card distributions
- Observe actions of each player (e.g. raise, fold)
- Probabilities for chance nodes are embedded in P_Φ





Featurized Rock Paper Scissors

$P =$

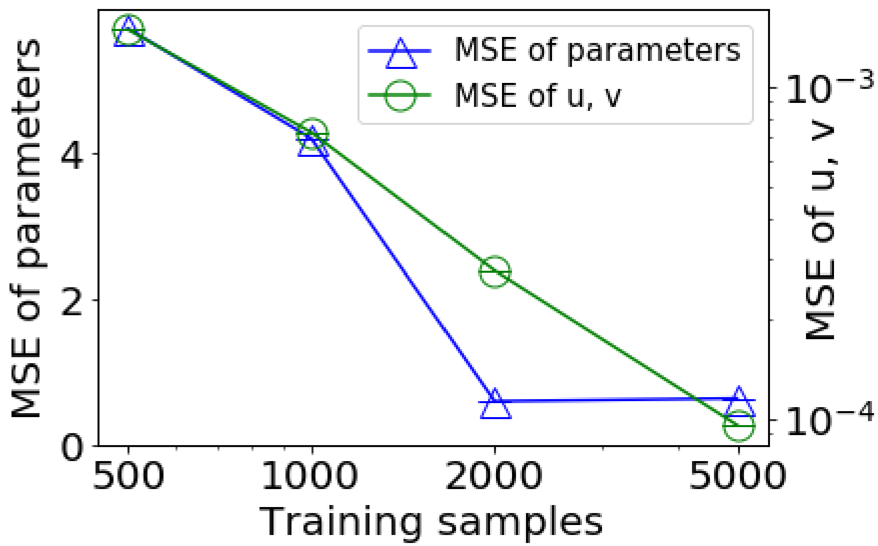
	R	P	S
R	0	$-b_1$	b_2
P	b_1	0	$-b_3$
S	$-b_2$	b_3	0

$b = \Phi x,$

$x \in [0,1]^2$

$\Phi \in [0,10]^{3\times 2}$

Objective is to learn Φ



Improve Scalability using FOM

- Recall in the basic approach, each step in the Newton's method of each forward pass requires solving a linear system → Time consuming

$$\begin{bmatrix} diag(\frac{1}{u}) & P & & \\ P^T & -diag\left(\frac{1}{v}\right) & 1 & 0 \\ & & 0 & 1 \\ 1^T & 0 & 0 & 0 \\ 0 & 1^T & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta u \\ \Delta v \\ \Delta \mu \\ \Delta \nu \end{bmatrix} = - \begin{bmatrix} Pv + \log(u) + 1 + \mu 1 \\ P^T u - \log(v) - 1 + \nu 1 \\ 1^T u - 1 \\ 1^T v - 1 \end{bmatrix}$$

- Solution: Use first-order iterative method (FOM) to solve the forward pass directly

$$\min_u \max_v u^T P v - \sum_i v_i \log v_i + \sum_i u_i \log u_i$$

$$1^T u = 1,\ 1^T v = 1$$

Large Scale Learning of Agent Rationality in Two-Player Zero-Sum Games. Chun Kai Ling, Fei Fang, Zico Kolter. In AAAI-19

Improve Scalability using FOM

- The problem in the forward pass is a problem of the following min-max format, where the last two terms are strictly convex functions

$$\min_{Ex=x_0} \max_{Fy=y_0} x^T P y + \mathcal{E}(x) - \mathcal{F}(y)$$

- This problem can be solved using various FOMs

Input: $x^{(0)}, y^{(0)}, P, \tau, \sigma$
for i *in* $\{0, \ldots\}$ **do**
$\quad \tilde{y} = y^{(i)}$;
$\quad x^{(i+1)} = \boxed{\mathrm{BR}_x(x^{(i)}, \tilde{y}; P, \tau);}$
$\quad \tilde{x} = 2x^{(i+1)} - x^{(i)}$;
$\quad y^{(i+1)} = \boxed{\mathrm{BR}_y(y^{(i)}, \tilde{x}; P, \sigma);}$
end

BR is smoothed best response

$$\mathrm{BR}_x(\bar{x}, \tilde{y}) = \arg\min_{Ex=x_0} x^T P \tilde{y} + \mathcal{E}(x) + \frac{1}{\tau} D_x(x, \bar{x})$$

$$\mathrm{BR}_y(\bar{y}, \tilde{x}) = \arg\min_{Fy=y_0} -\tilde{x}^T P y + \mathcal{F}(y) + \frac{1}{\sigma} D_y(y, \bar{y}).$$

Improve Scalability using FOM

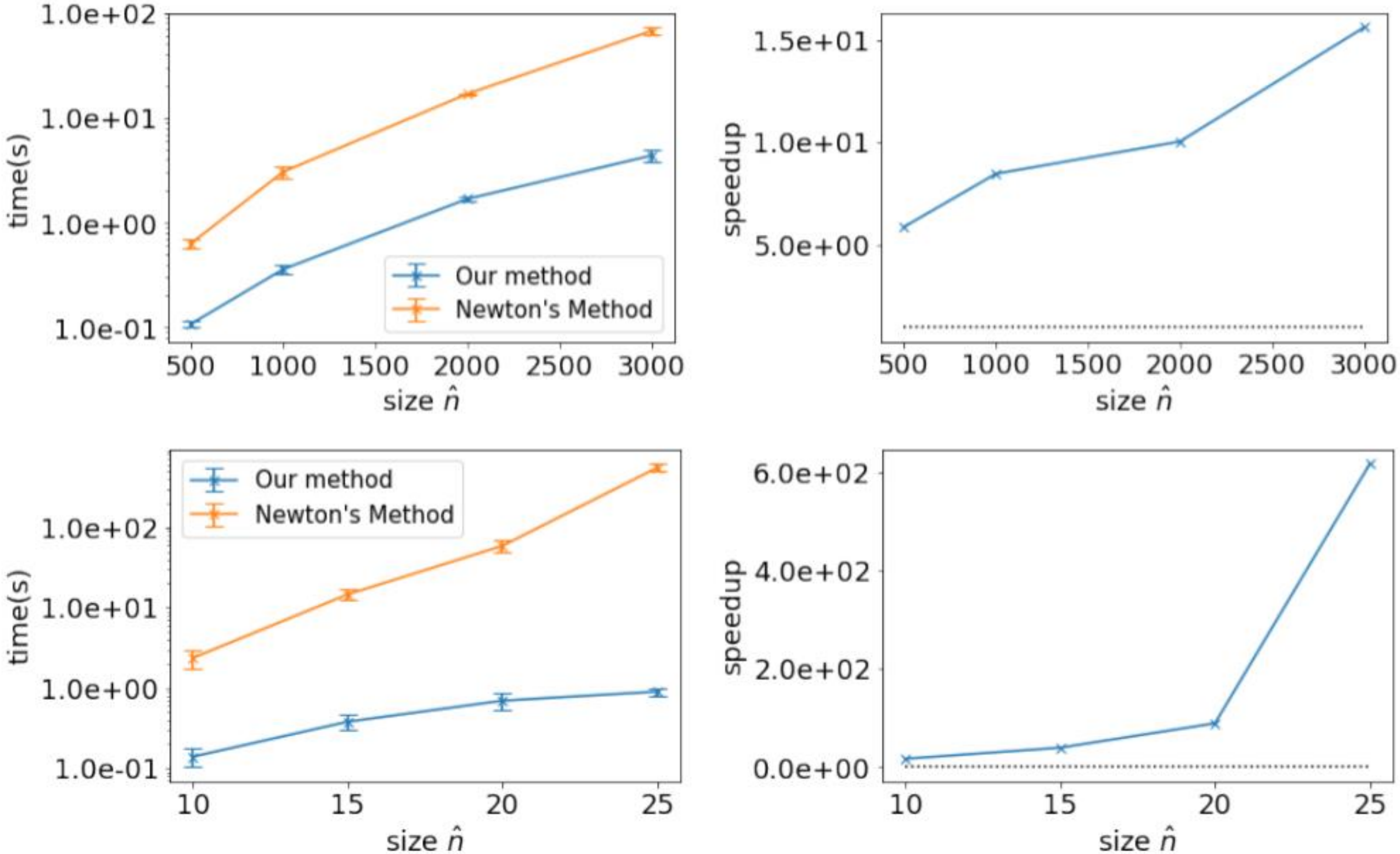
- Surprisingly, solving each step in the backward pass can also be converted to solving a problem with the min-max format. So same FOM can be applied.

$$\begin{bmatrix} y_u \\ y_v \\ y_\mu \\ y_\nu \end{bmatrix} = \begin{bmatrix} -\Xi(u) & P & E^T & 0 \\ P^T & \Xi(v) & 0 & F^T \\ E & 0 & 0 & 0 \\ 0 & F & 0 & 0 \end{bmatrix}^{-1} \begin{bmatrix} -\nabla_u L \\ -\nabla_v L \\ 0 \\ 0 \end{bmatrix}$$

$$\uparrow \text{KKT Conditions}$$

$$\begin{aligned} \min_x \max_y \quad & x^T P y + \frac{1}{2} x^T \Xi(u) x - \frac{1}{2} y^T \Xi(v) y \\ & + \nabla_u L^T x + \nabla_v L^T y \\ \text{subject to} \quad & Ex = 0 \qquad Fy = 0. \end{aligned}$$

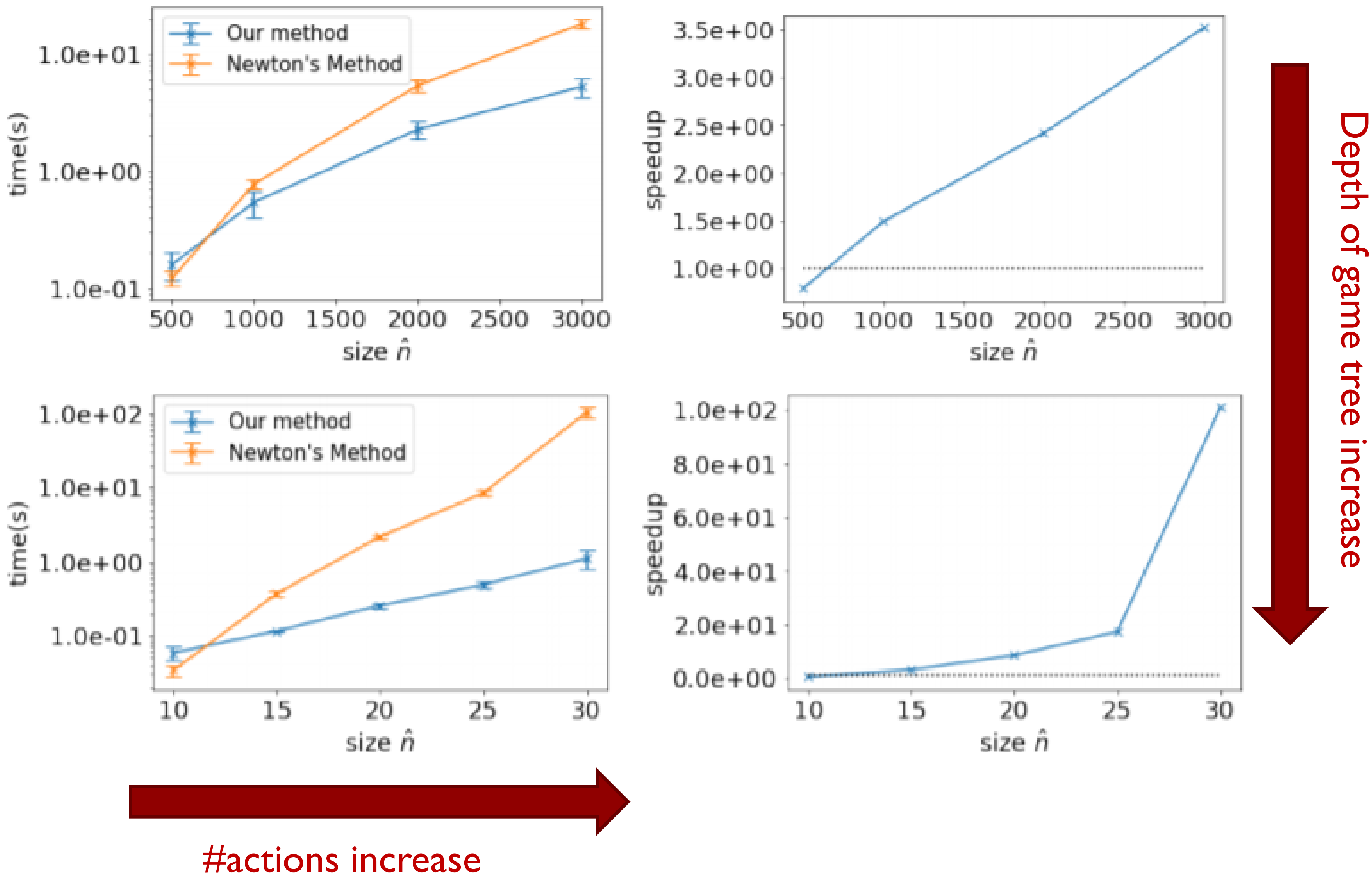
Speedup in Forward Pass



Depth of game tree increase

#actions increase

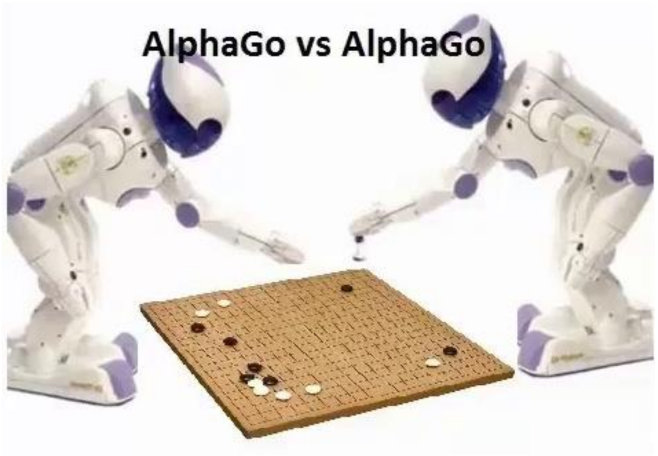
Speedup in Backward Pass



Outline

- Games with Human Players for Real-world Applications
  - Wildlife Conservation
- End-to-End Learning and Decision Making in Games
  - A differentiable learning framework for learning game parameters
- Learning-Powered Strategy Computation in Large Games
  - Leveraging Deep Reinforcement Learning
- Other Applications and Summary

Solving Game through Learning from Self Play



https://www.youtube.com/watch?v=Ue4A2Y_i3ZQ

More Complex Games: Patrol with Real-Time Information

- Sequential interaction
  - Players make flexible decisions instead of sticking to a plan
  - Players may leave traces as they take actions

- Example domain: Wildlife protection

Footprints

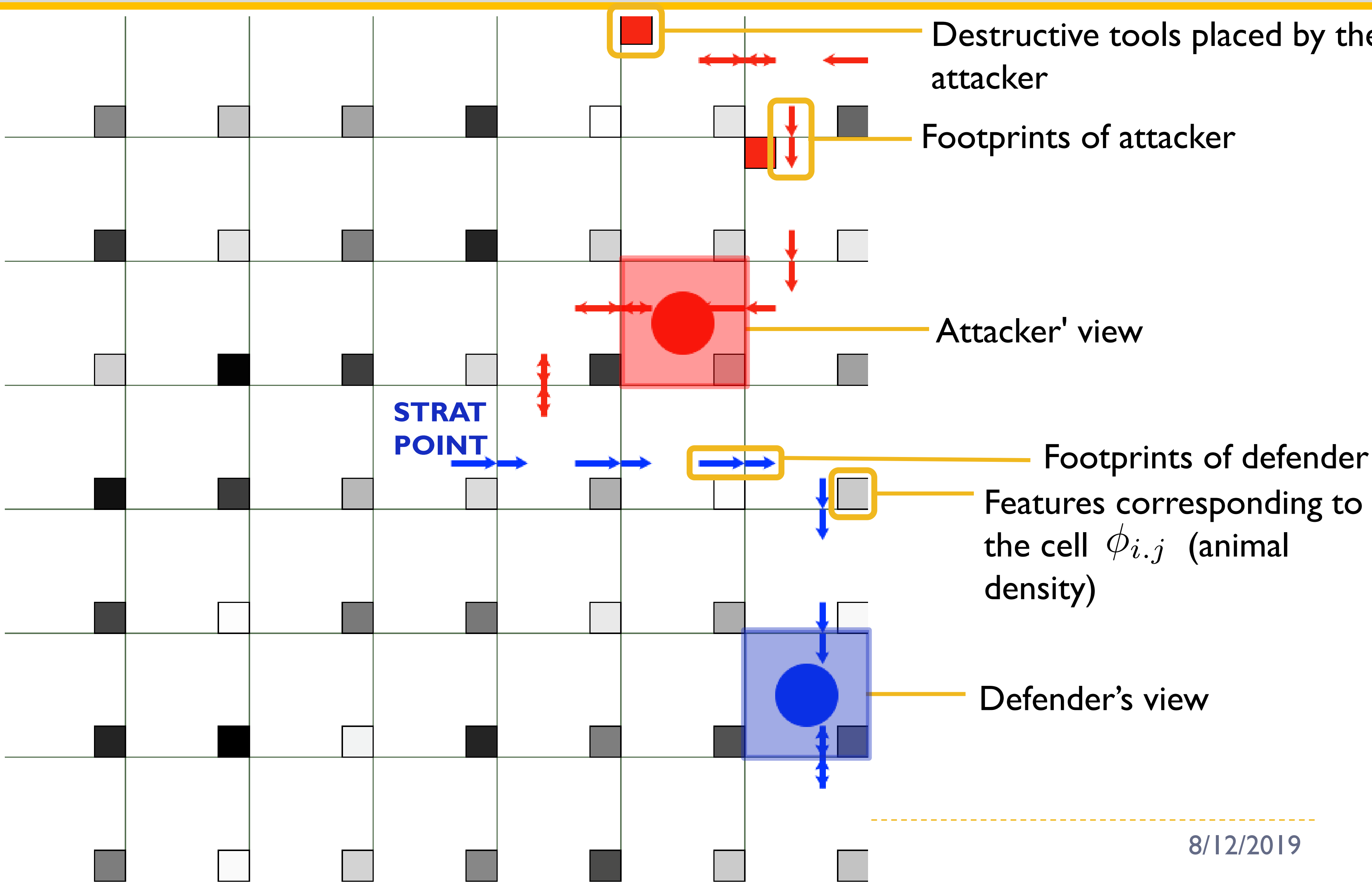
Lighters

Old poacher camp

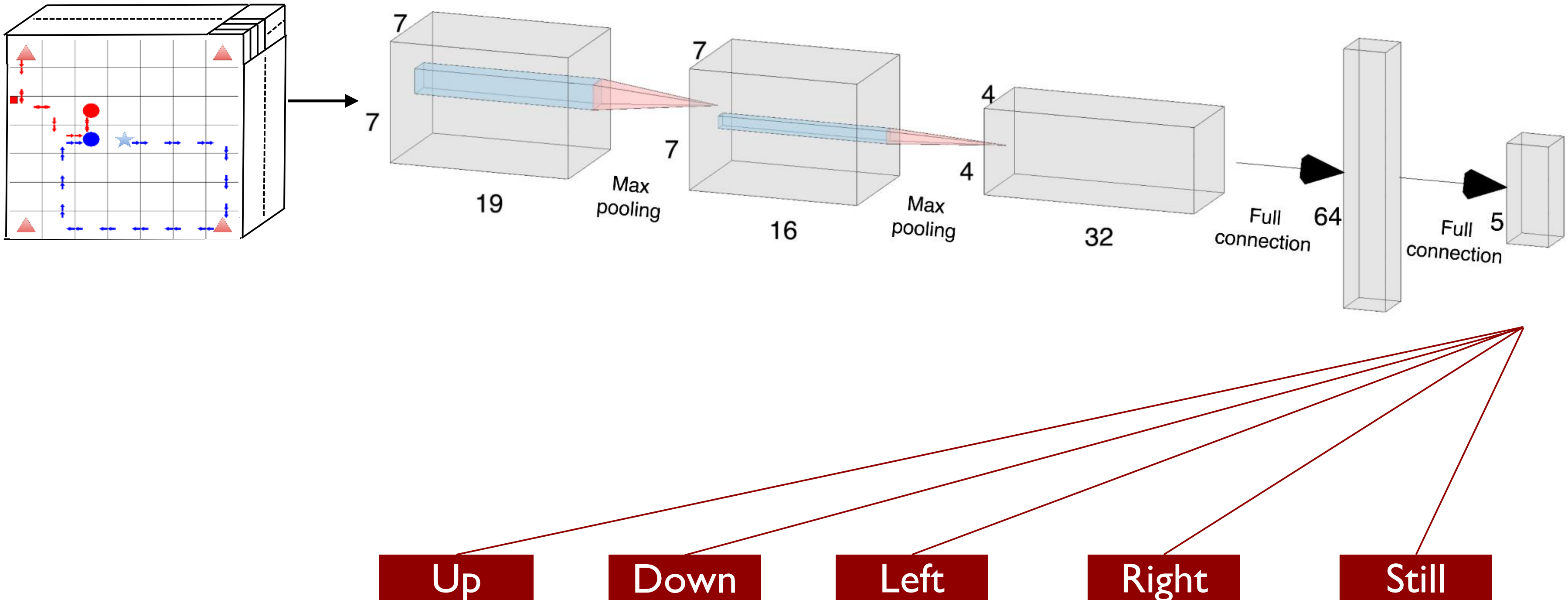
Tree marking

Deep Reinforcement Learning for Green Security Games with Real-Time Information Yufei Wang, Zheyuan Ryan Shi, Lantao Yu, Yi Wu, Rohit Singh, Lucas Joppa, Fei Fang In AAAI-19

Multi-Agent Reinforcement Learning

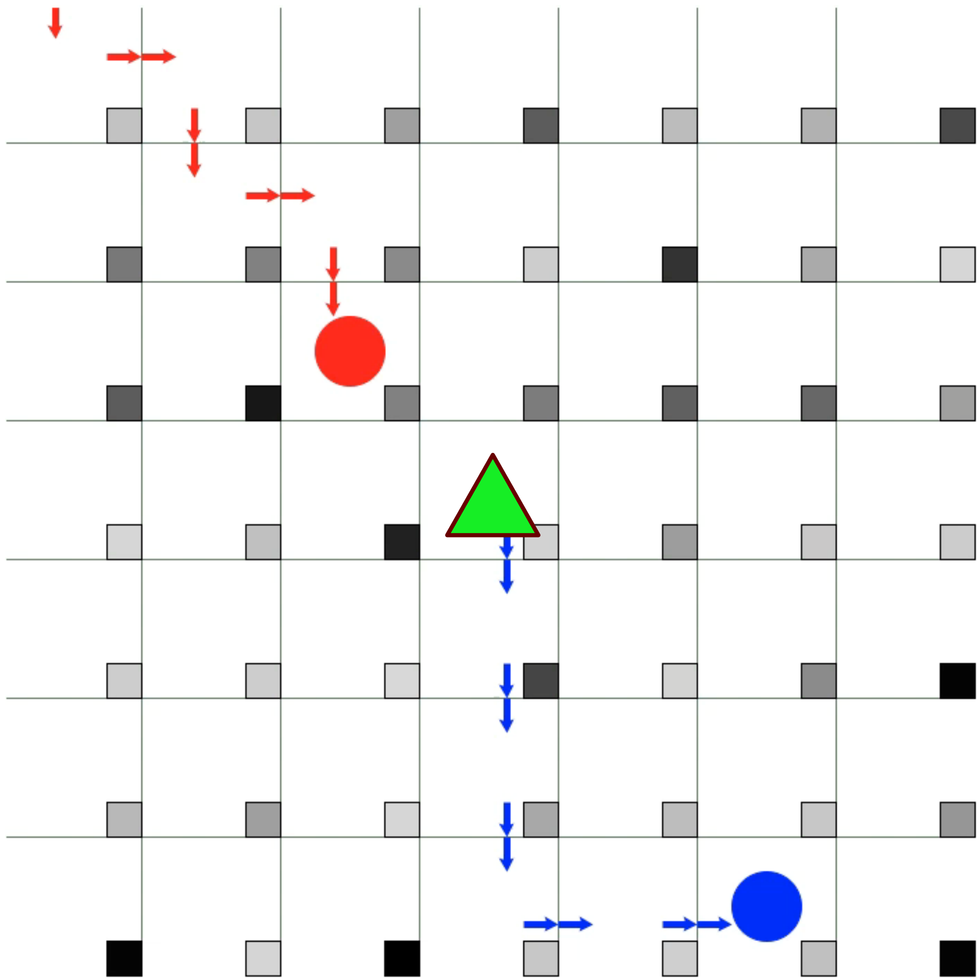


Compute Best Response by Training a Deep Q-Network



- Q Network: Game state → Q-value
- Use Deep Reinforcement learning to train the network and find optimal patrol policy (assuming fixed attacker)

Compute Best Response by Training a Deep Q-Network



DQN Defender
vs
Non-Adaptive Attacker

Attacker

Snares

Start from one of the corners

Defender

Start from
Patrol Base

Compute Equilibrium: DQN + Double Oracle

Compute $\sigma^d, \sigma^a = Nash(G^d, G^a)$

Compute Nash / Minimax

Train $f^d = DQN(\sigma^a)$

Find Best Response to attacker's strategy

Train $f^a = DQN(\sigma^a)$

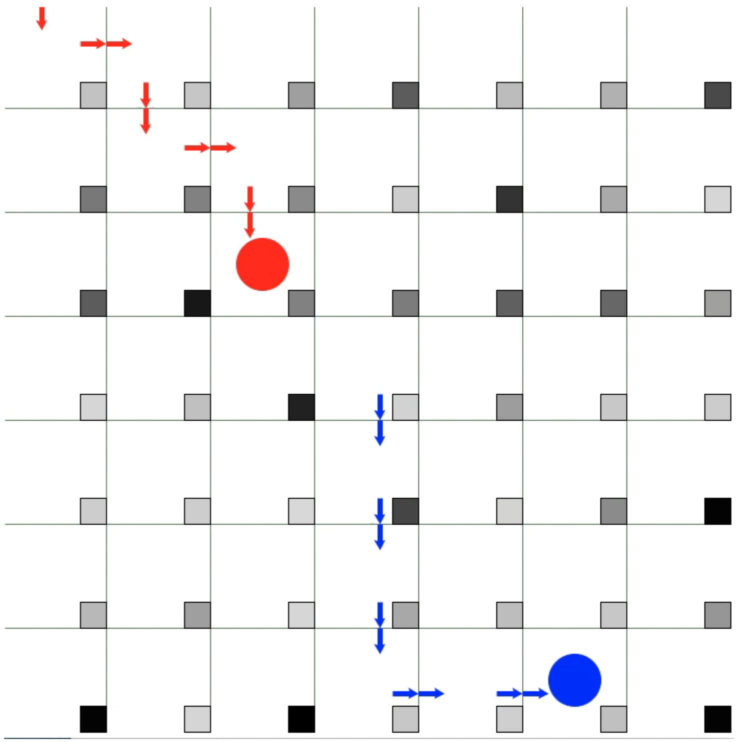
Find Best Response to defender's strategy

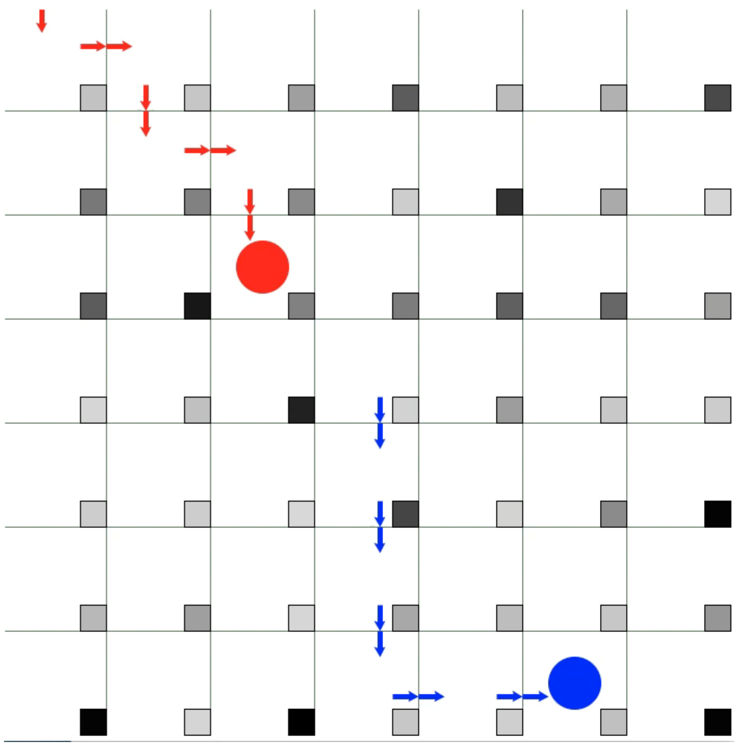
Add f^d, f^a to G^d, G^a

Update basic strategy set

Enhancements

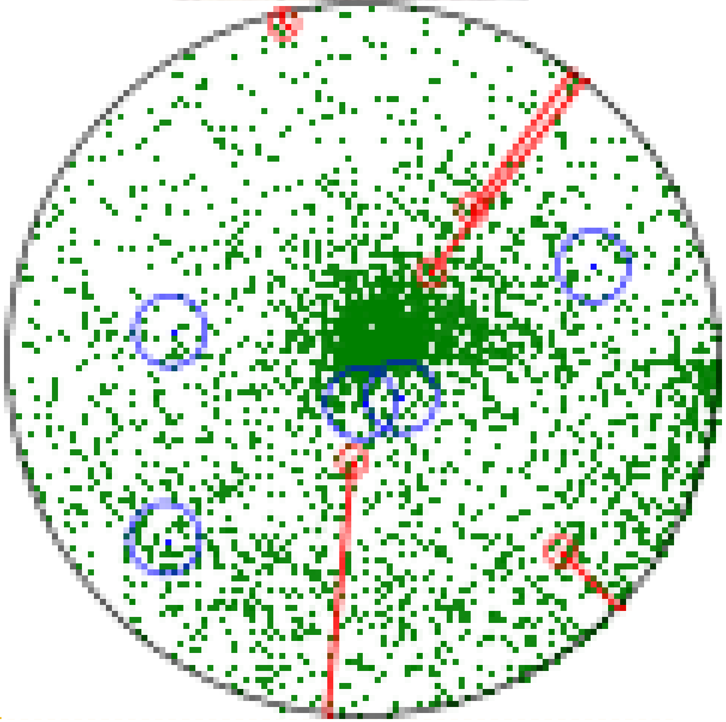
- Use local modes for efficient and parallized training
- Start with domain-specific heuristic strategies





Solving Game through Learning from Self Play

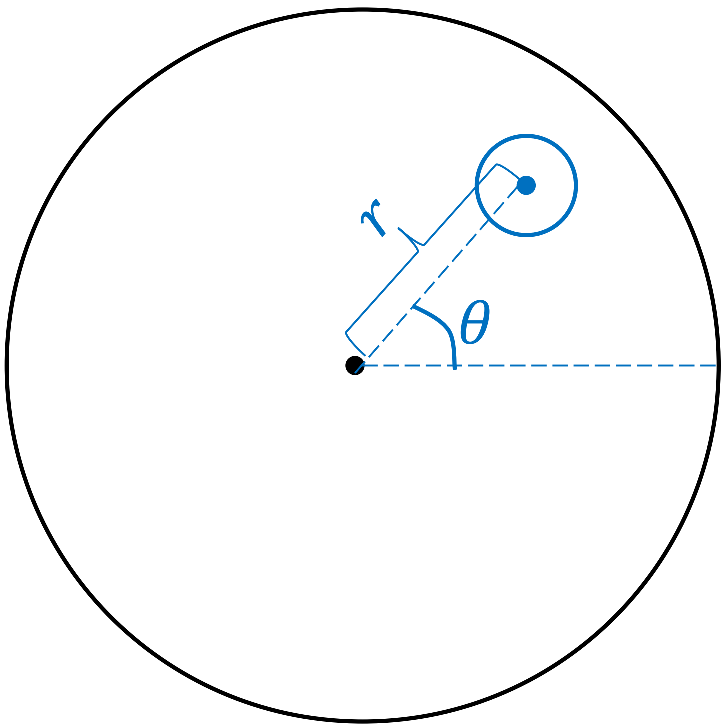
- Green dots: Valuable trees
- Blue dots: Defender location
- Red dots: Logging locations
- Zero-sum game
- Goal: Find defender strategy or defender policy



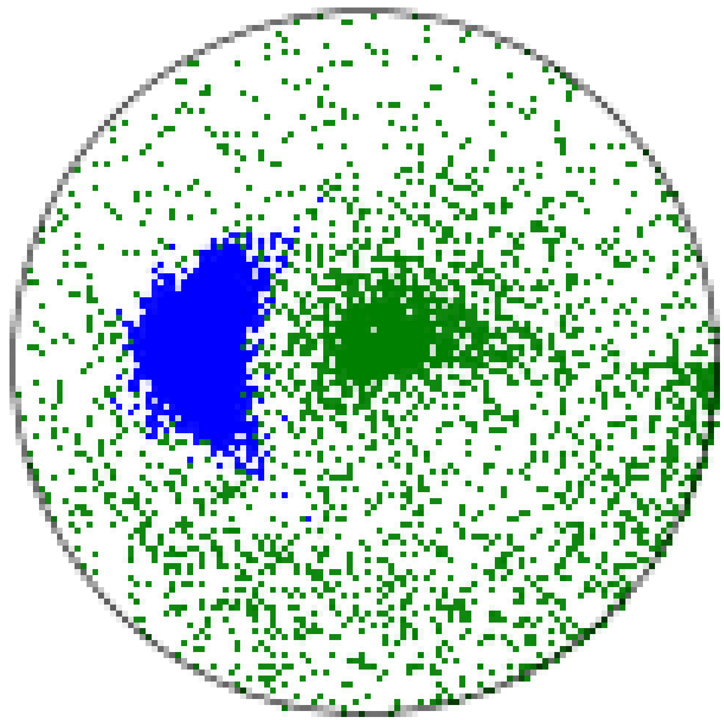
Policy Learning for Continuous Space Security Games using Neural Networks. Nitin Kamra, Umang Gupta, Fei Fang, Yan Liu, Milind Tambe. In AAAI-18

Solving Game through Learning from Self Play

- Key idea 1: Represent mixed strategy using logit normal distribution in polar coordinate system

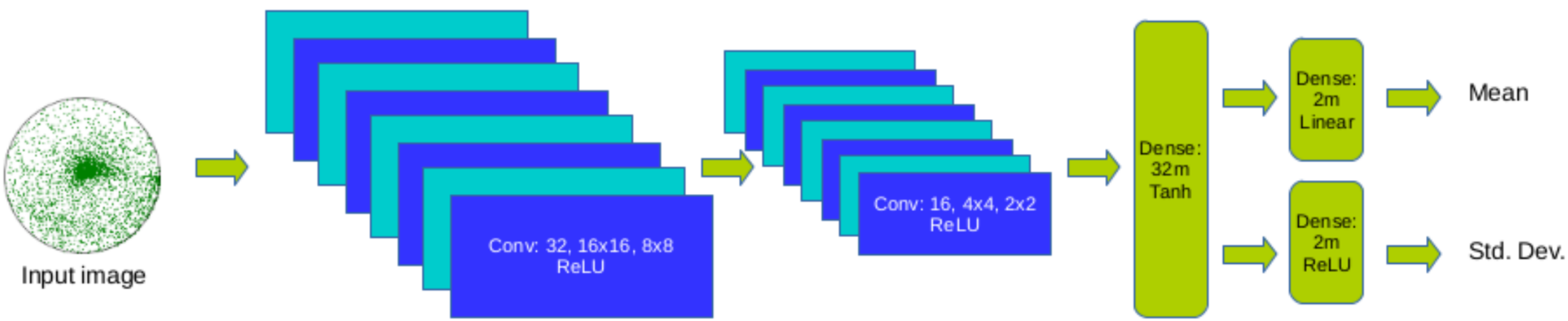


$$r \sim P(\mathcal{N}(\mu_r, \sigma_r^2))$$
$$\theta \sim P(\mathcal{N}(\mu_\theta, \sigma_\theta^2))$$



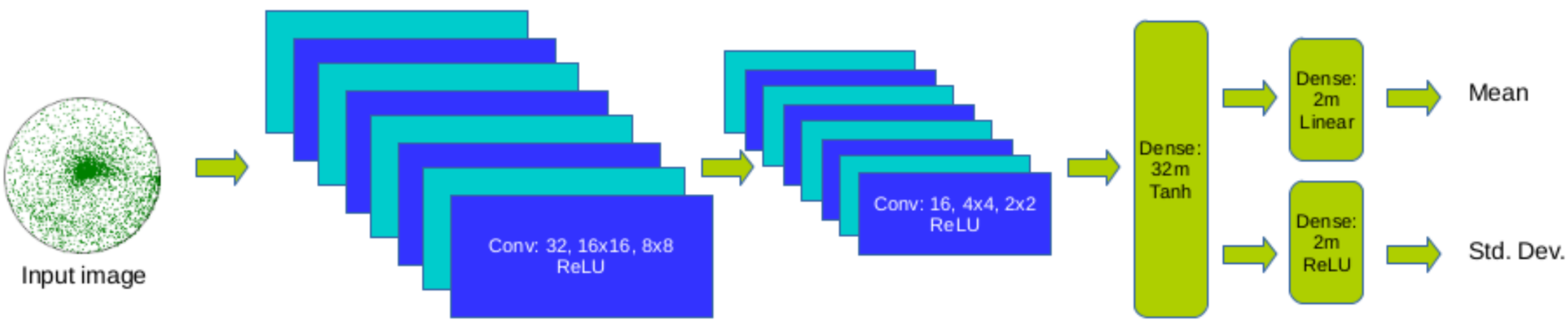
Solving Game through Learning from Self Play

- Key idea 2: Represent a "policy" with Convolutional Neural Network
  - Policy: mapping from game setting to strategy
  - CNN: Tree Distribution →Mean/Std of r and θ



Solving Game through Learning from Self Play

- Key idea 3: Approximate Fictitious Play
  - Fictitious Play: Best responds to opponent's average strategy
  - Average strategy → Random samples from history
  - Best response → Update neural network



Solving Game through Learning from Self Play

- Put them together

Algorithm 1: OptGradFP

Initialization. Initialize policy parameters $\boldsymbol{w_D}$ and $\boldsymbol{w_O}$, replay memory *mem*;

for *ep in* $\{0, \ldots, ep_{max}\}$ **do**

  Simulate n_s game play. Sample game setting and actions from current policy π_D and π_O n_s times, save in *mem*;

  Replay for defender. Draw n_b samples from *mem*, resample defender action from current policy π_D;

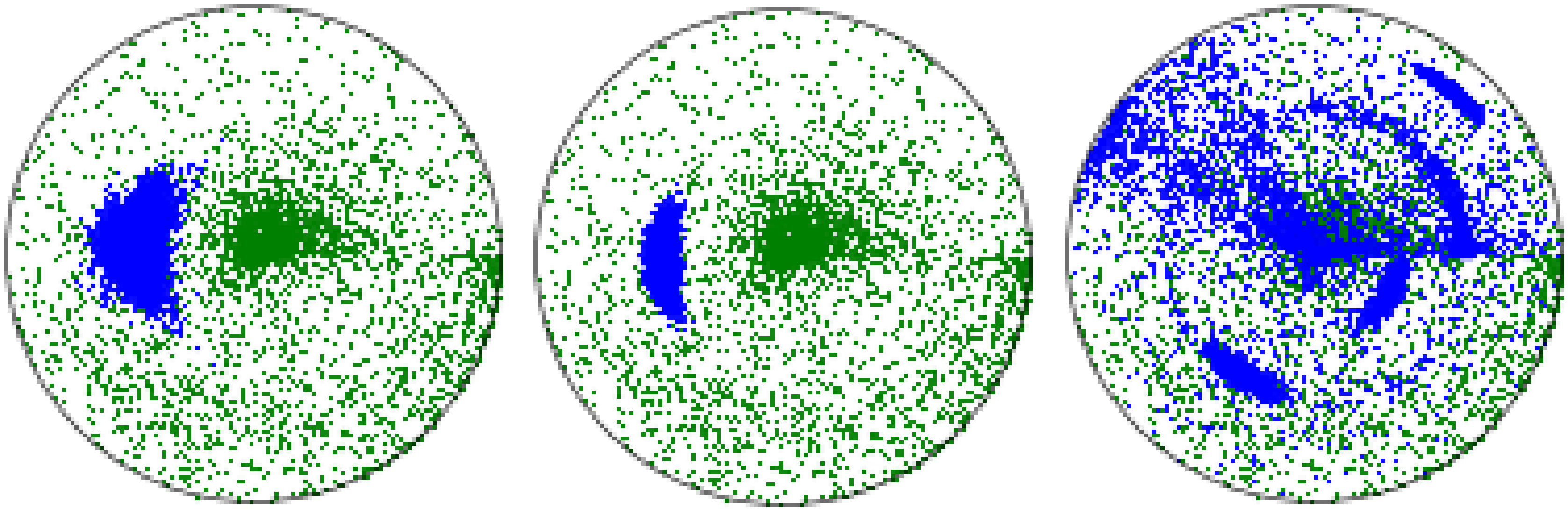
  Update parameter for defender. Update defender policy parameter $\boldsymbol{w_D} := \boldsymbol{w_D} + \frac{\alpha_D}{1+ep\,\beta_D} * \nabla_{\boldsymbol{w_D}} J_D$;

  Replay for attacker. Draw n_b samples from *mem*, resample attacker action from current policy π_O;

  Update parameter for attacker. Update attacker policy parameter $\boldsymbol{w_O} := \boldsymbol{w_O} + \frac{\alpha_O}{1+ep\,\beta_O} * \nabla_{\boldsymbol{w_O}} J_O$

Solving Game through Learning from Self Play

- Single game setting



Cournot Adjustment StackGrad OptGradFP

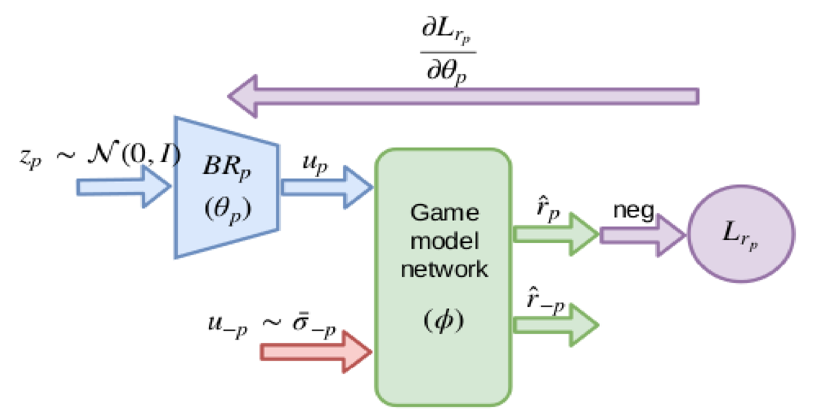
- Multiple game setting
  - Train on 1000 forest states, predict on unseen forest state
  - 7 days for training, Prediction time 90 ms
  - Shift computation from online to offline

Enhancement

- DeepFP
  - Generative network for approx. BR + game model network
  - Allow to use mathematical programming-based approach to compute BR for one or both players

Data: max_games, batch sizes (m_1, m_2, m_G), memory size E, game simulator and oracle BRO_p for players with no gradient
Result: Final belief densities $\bar{\sigma}_p^*$ in mem $\forall$ players p
Initialize all network parameters $(\theta_1, \theta_2, \phi)$ randomly;
Create empty memory mem of size E;
for $game \in \{1, \ldots, max\_games\}$ **do**

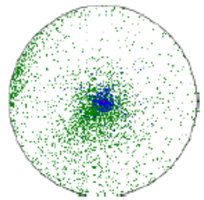
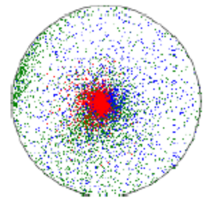
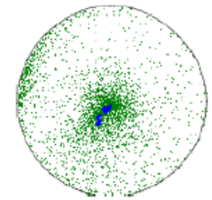
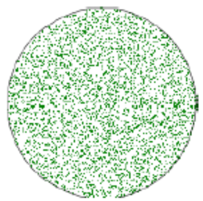
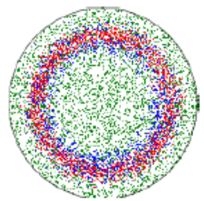
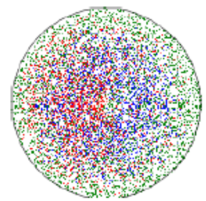
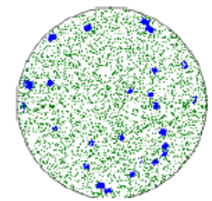
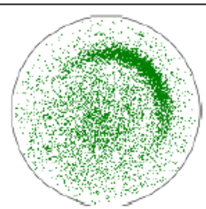
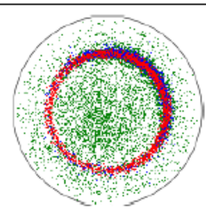
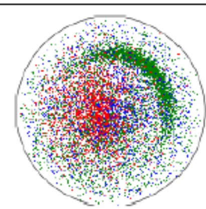
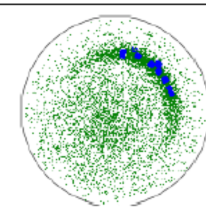
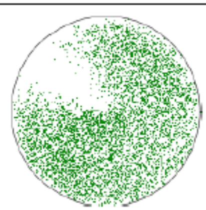
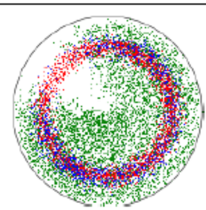
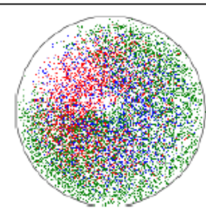
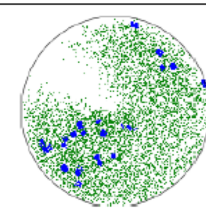
```
    Obtain best responses
    Play game and update memory
    Train shared game model net
    Train best response nets
```



$$L_{r_p}(\theta_p) = -\mathbb{E}_{(z_p \sim \mathcal{N}(0,I), u_{-p} \sim \bar{\sigma}_{-p})}[\hat{r}_p(BR_p(z_p; \theta_p), u_{-p}; \phi)]$$

DeepFP for Finding Nash Equilibrium in Continuous Action Spaces. Nitin Kamra, Umang Gupta, Kai Wang, Fei Fang, Yan Liu, Milind Tambe. In GameSec-19

Enhancement

Forest structure	DeepFP	OptGradFP	DLP (approx. ground truth)
 F1	 $\epsilon = 10.96 \pm 6.19$	 $\epsilon = 21.72 \pm 4.47$	 $\epsilon = 12.48 \pm 2.23$
 F2	 $\epsilon = 1.20 \pm 0.07$	 $\epsilon = 1.28 \pm 0.07$	 $\epsilon = 0.53 \pm 0.10$
 F3	 $\epsilon = 8.96 \pm 3.65$	 $\epsilon = 14.58 \pm 0.20$	 $\epsilon = 0.49 \pm 0.13$
 F4	 $\epsilon = 2.05 \pm 0.13$	 $\epsilon = 2.18 \pm 0.09$	 $\epsilon = 0.12 \pm 0.07$

Outline

- Games with Human Players for Real-world Applications
  - Wildlife Conservation
- End-to-End Learning and Decision Making in Games
  - A differentiable learning framework for learning game parameters
- Learning-Powered Strategy Computation in Large Games
  - Leveraging Deep Reinforcement Learning
- Other Applications and Summary

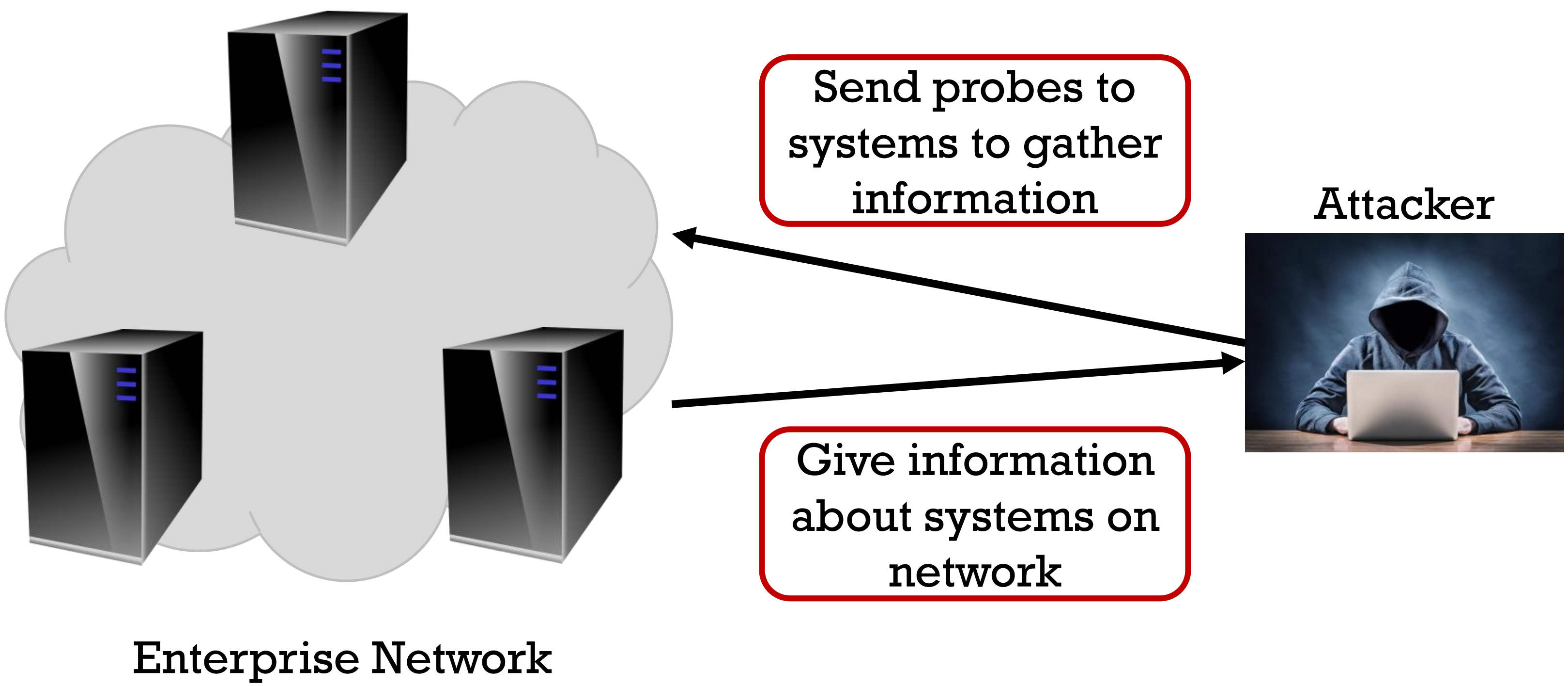
How Valuable is This Car?

Deception

Deception

Cyber Deception

- What can the defender do without "patrol boats"?
- Use deception to confuse the attackers!



Deceiving Cyber Adversaries: A Game Theoretic Approach. Aaron Schlenker, Omkar Thakoor, Haifeng Xu, Fei Fang, Milind Tambe, Long Tran-Thanh, Phebe Vayanos, Yevgeniy Vorobeychik. In AAMAS-18

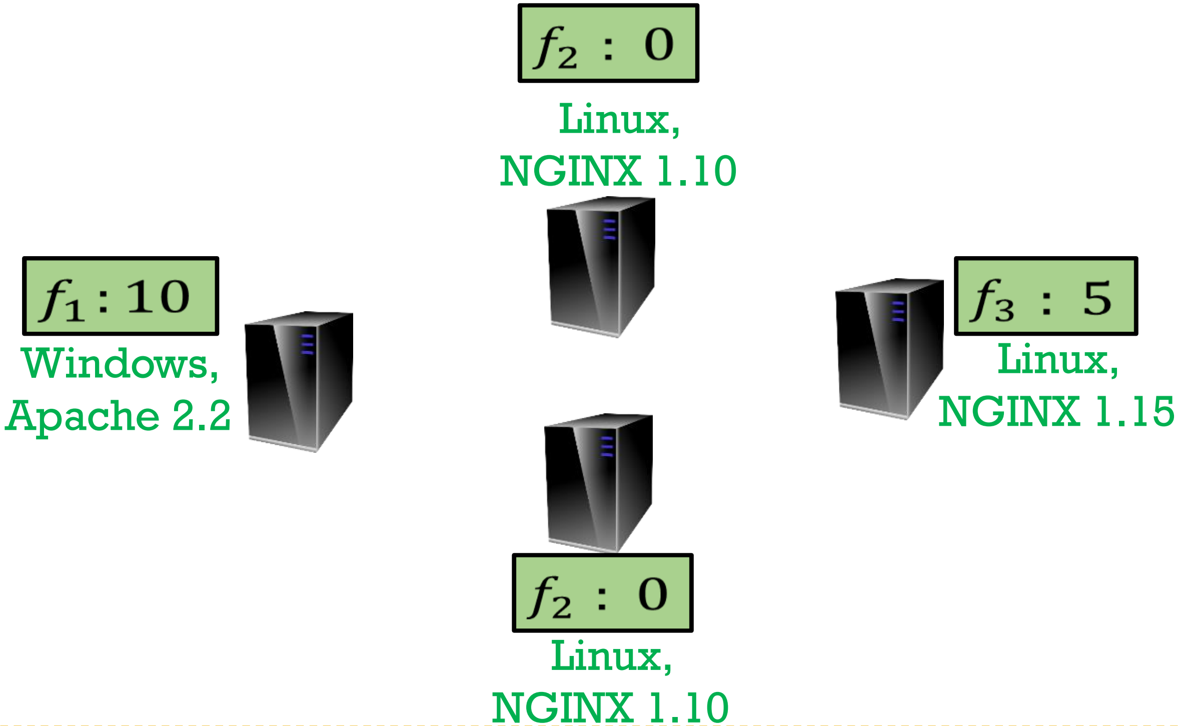
Cyber Deception

- How should the defender disguise the systems to induce the adversary to attack the least valuable systems?

- Cyber Domain Challenges:
  - Intelligent adversary; could perceive deception occurring
  - Large number of system configurations and ways to disguise
  - Arbitrary deception may not be feasible or may affect performance

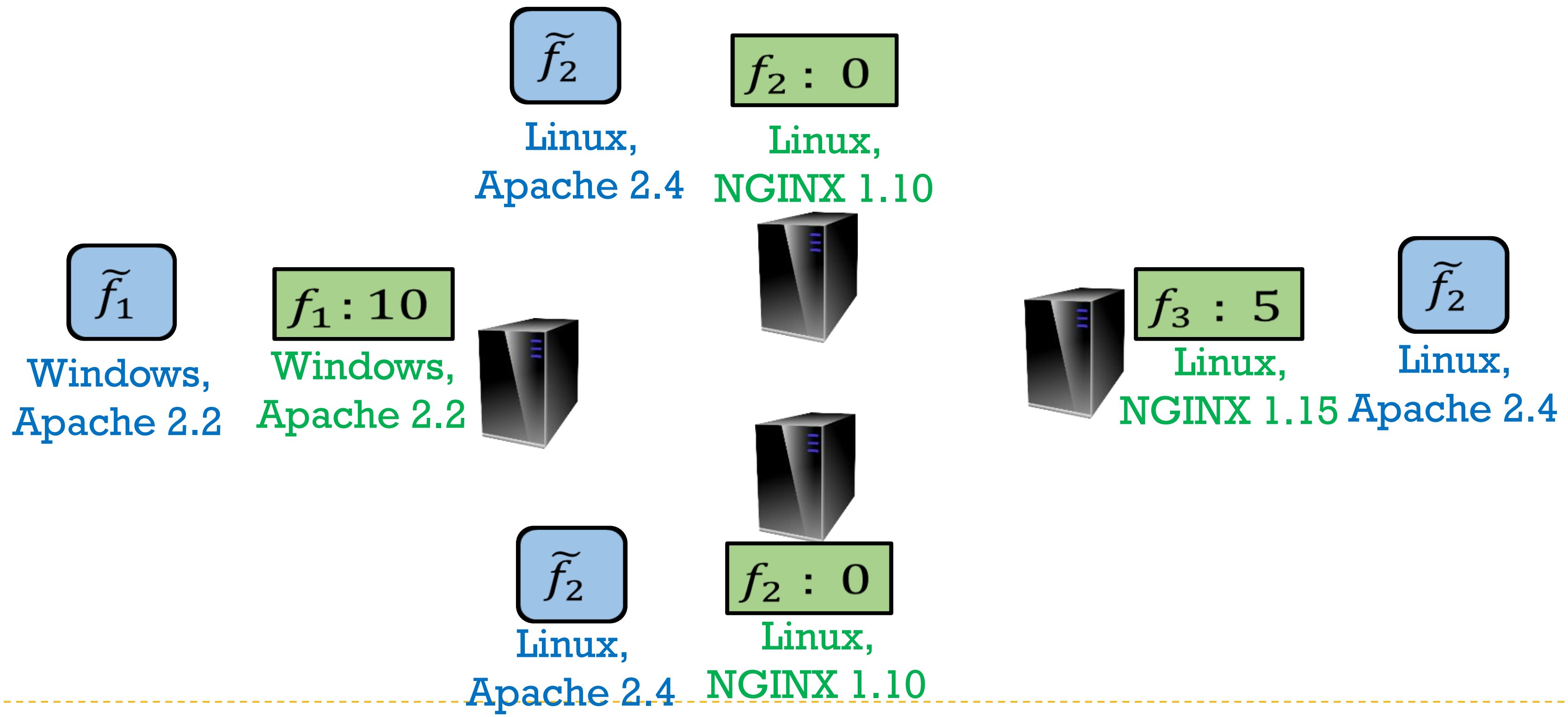
Cyber Deception Game: Setting

- K systems, each has true configuration (TC) $f \in F$
- Successful attack on system with TC f yields utility U_f to attacker; defender loses U_f (gains $-U_f$)



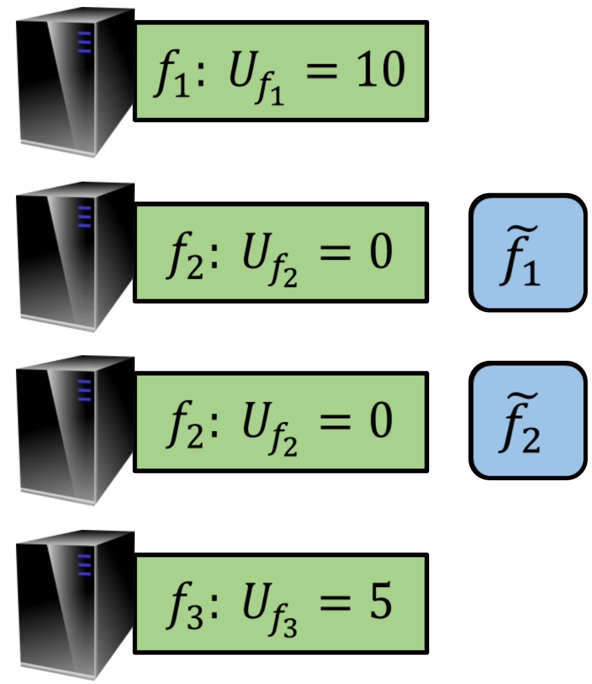
Cyber Deception Game: Setting

- Defender disguise the systems through deceptive responses
- Each system gets observed configuration (OC) $\tilde{f} \in \tilde{F}$



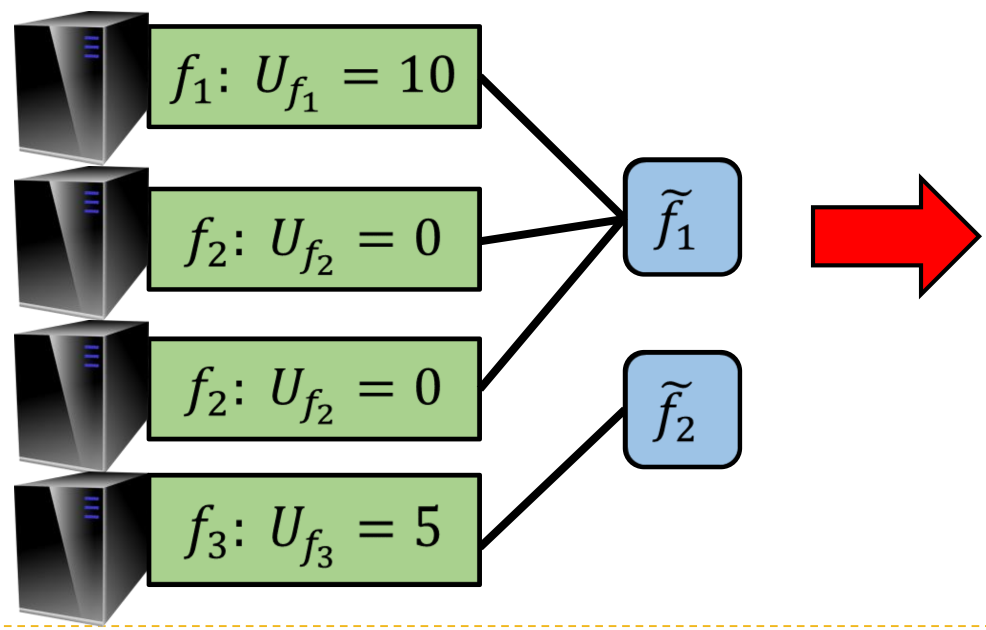
Cyber Deception Game: Defender

- Know true configuration (TC) f
- Need to decides observed configuration (OC) $\tilde{f}$
- Systems with same TC are indifferent to the defender
- N_f = Number of systems having TC $f \in F$



Cyber Deception Game: Defender

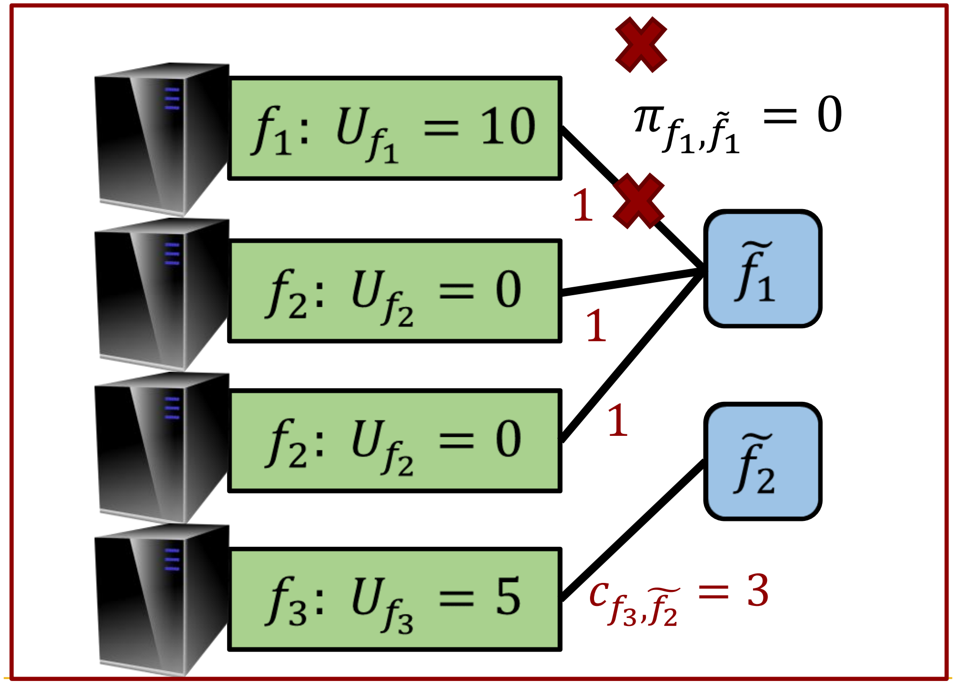
- Deception strategy encoded via integer matrix ϕ
  - $\phi_{f,\tilde{f}}$ = number of systems with TC f and OC $\tilde{f}$



$\boldsymbol{\phi}$	$\tilde{f}_1$	$\tilde{f}_2$
f_1	1	0
f_2	2	0
f_3	0	1

Cyber Deception Game: Defender

- Deception strategy encoded via integer matrix ϕ
  - $\phi_{f,\tilde{f}}$ = number of systems with TC f and OC $\tilde{f}$
  - TC f may not be masked with OC $\tilde{f}$ ($\pi_{f,\tilde{f}} = 0$)
  - Showing deceptive responses incur costs $c(f,\tilde{f})$; budget B



$B = 5$

Cyber Deception Game: Attacker

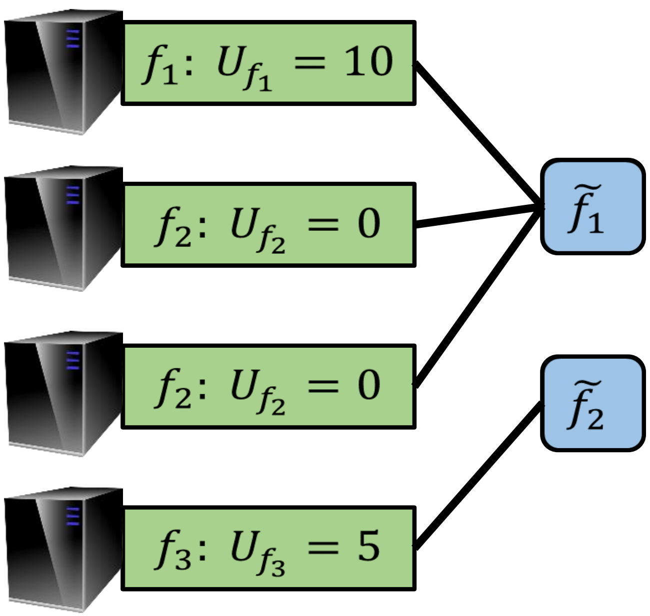
- Can observe OC of each system
- Cannot differentiate systems with same OC
- Uniformly randomly attacks systems with **most attractive** OC

↓

How much does the attacker know about the deception?

Cyber Deception Game: Attacker

- Powerful attacker: Knows deception strategy ϕ
  - Computes expected payoff for all OCs and best-responds
  - Robust assumption to minimize worst-case loss



$\boldsymbol{\phi}$	$\tilde{f}_1$	$\tilde{f}_2$
f_1	1	0
f_2	2	0
f_3	0	1

Expected Payoff

$$\tilde{U}_{\tilde{f}} = \frac{\sum_{f\in F} \phi_{f,\tilde{f}} U_f}{\sum_{f\in F} \phi_{f,\tilde{f}}}$$

$$\tilde{U}_{\tilde{f}_1} = \frac{10 + 2 * 0}{3} = 3.33$$

$$\tilde{U}_{\tilde{f}_2} = 5/1 = 5$$

Cyber Deception Game: Attacker

- Powerful attacker: Knows deception strategy ϕ
  - Computes expected payoff for all OCs and best-responds
  - Robust assumption to minimize worst-case loss

- Naive attacker: Not aware of deception
  - Believe what they observe
  - Preset preferences (utilities) for attacking OCs

Quiz

- With powerful attacker, when there are no budget constraint and feasibility constraint, what is the optimal defender strategy?

Quiz

- With powerful attacker, when there are no budget constraint and feasibility constraint, what is the optimal defender strategy?

- Trivial case (no constraints): assign to same OC

Against Powerful Attacker

- Powerful attacker: Knows deception strategy ϕ
  - Computes expected payoff for all OCs and best-responds
  - Robust assumption to minimize worst-case loss

- When some masking infeasible or budget limited

<u>Theorem</u>: NP-hard to compute optimal strategy for defender against powerful adversary.

  - Proven via reduction to Partition problem
  - NP-hard even with just feasibility or just budget constraint

Against Powerful Attacker

- Solve through mathematical programming

Non-linear

$$\min_{u,\phi} \quad u$$

$$s.t. \quad u \geq \boxed{\frac{\sum_{f\in F} \phi_{f,\tilde{f}} U_f}{\sum_{f\in F} \phi_{f,\tilde{f}}}} \qquad \forall \tilde{f} \in \tilde{F}$$

Expected Utility for attacking $\tilde{f}$

$$\sum_{\tilde{f}} \phi_{f,\tilde{f}} = N_f$$

$$\sum_{f} \phi_{f,\tilde{f}} = N_{\tilde{f}}$$

$$\phi_{f,\tilde{f}} \leq \pi_{f,\tilde{f}}$$

$$\phi_{f,\tilde{f}} \in \mathbb{Z}_{\geq 0}$$

Feasibility Constraints

$$\sum_{f} \sum_{\tilde{f}} \phi_{f,\tilde{f}} c_{f,\tilde{f}} \leq B$$

Budget Constraint

Against Powerful Attacker

- Solve through mathematical programming
- Reformulate to MILP: Guaranteed to find optimal solution
  - Remove the non-linear constraint
  - Adds $|K||\tilde{F}|$ auxiliary variables
  - Adds $4|K||\tilde{F}|$ additional constraints
- Approximation algorithm: Solve sequential MILPs
- Heuristic algorithm: Greedy MiniMax (GMM)
  - A fast heuristic which greedily minimizes attacker utility

Against Naïve Attacker

- Naive attacker: Not aware of deception
  - Simply believes OCs (or just not reasoning about the actual TC→OC mapping strategy used by the defender)
  - Preset preferences (utilities) for attacking OCs
- When no budget constraints; but just the feasibility constraints

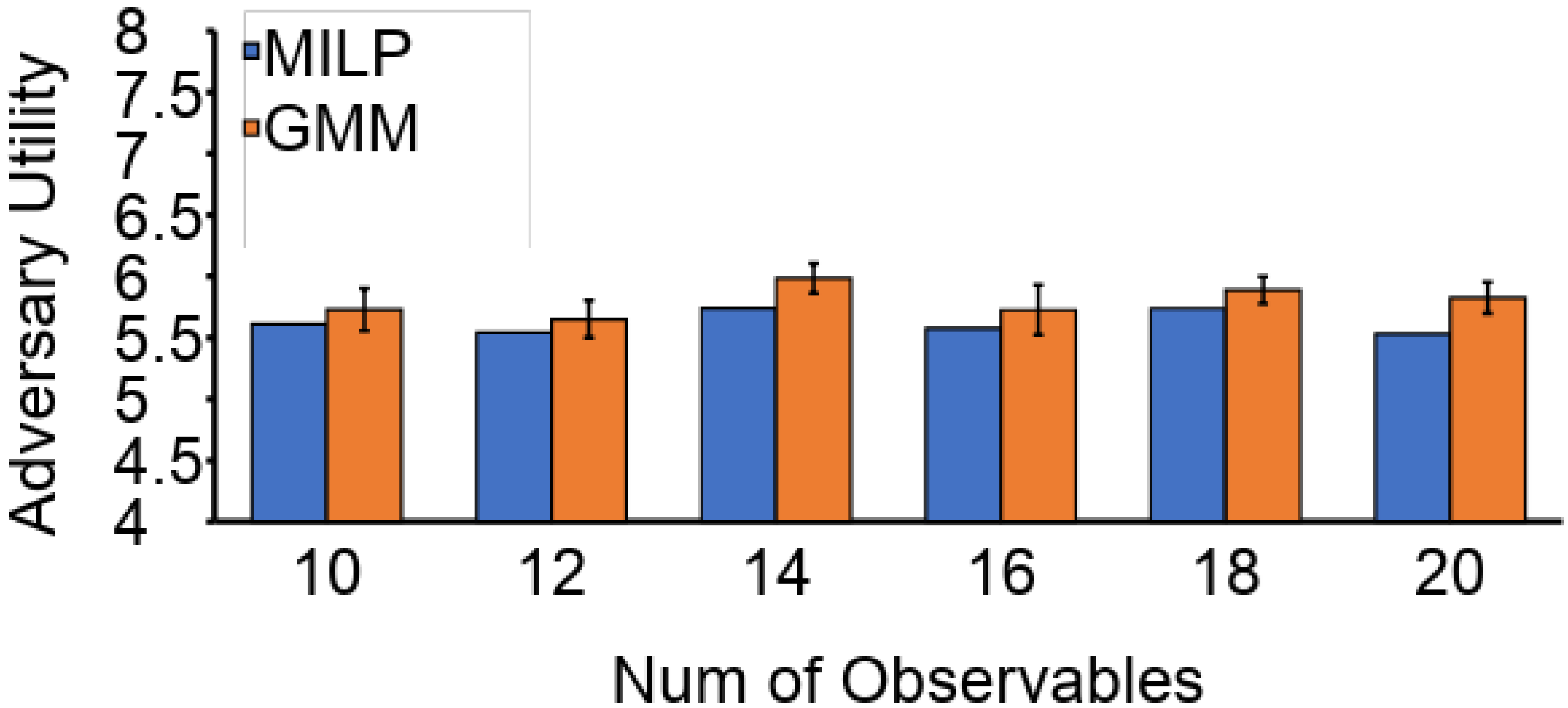
Theorem: can be solved in $O(|F||\tilde{F}|)$ time

- When both budget and feasibility constraints present

Theorem: NP-hard to compute optimal strategy for defender against naïve adversary.

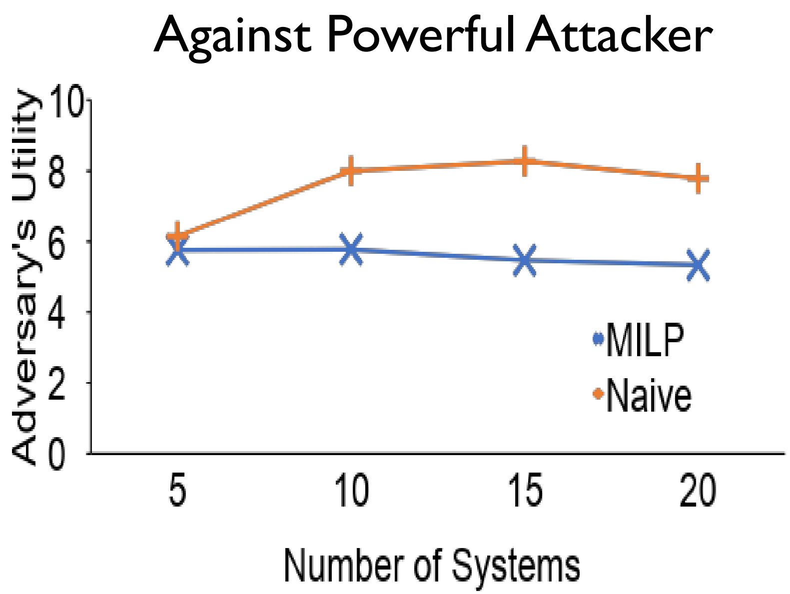
Simulation Results

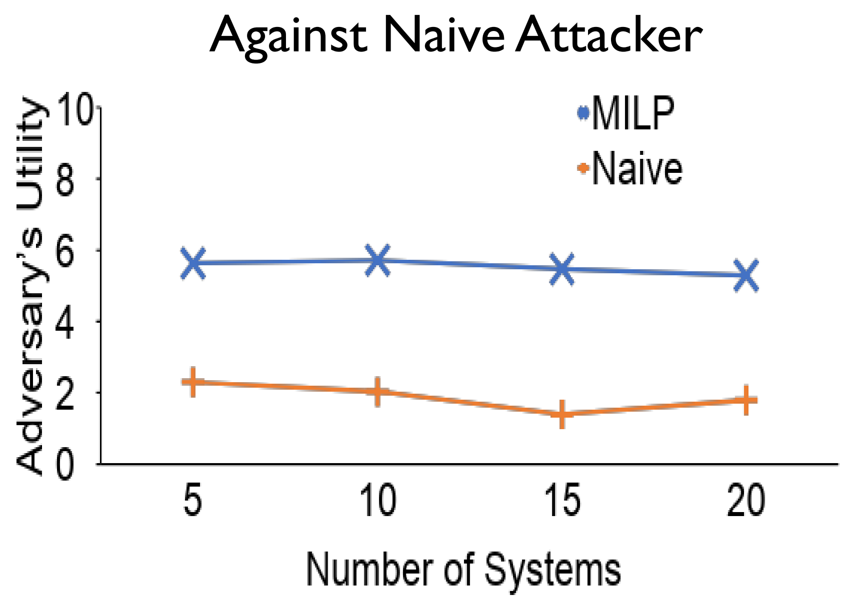
- 20 TCs, 20 Systems
- Attacker Utility = 10 without deception



Simulation Results

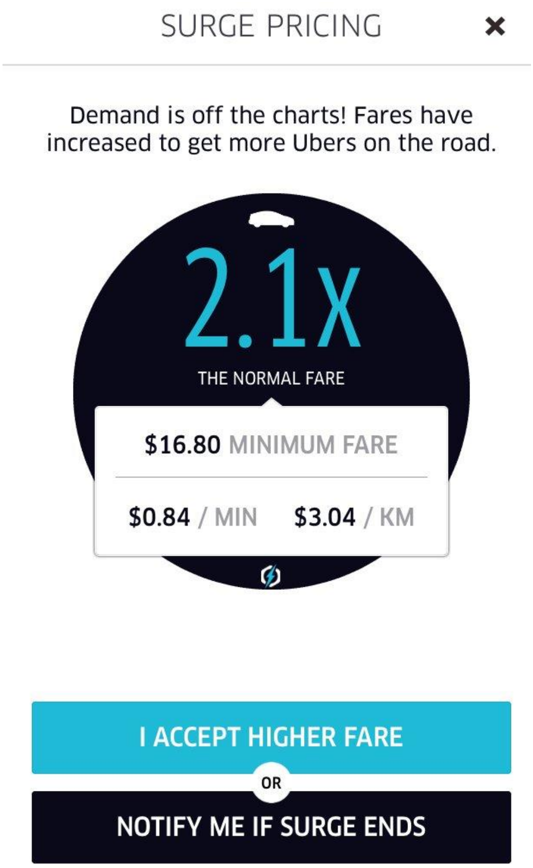
- Attacker model and belief of attacker model matters

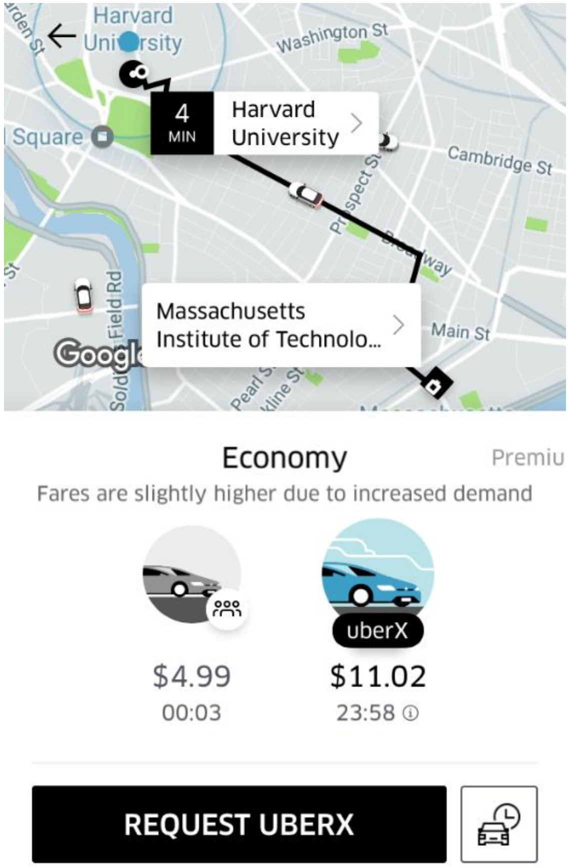




Evolution of Surge Pricing

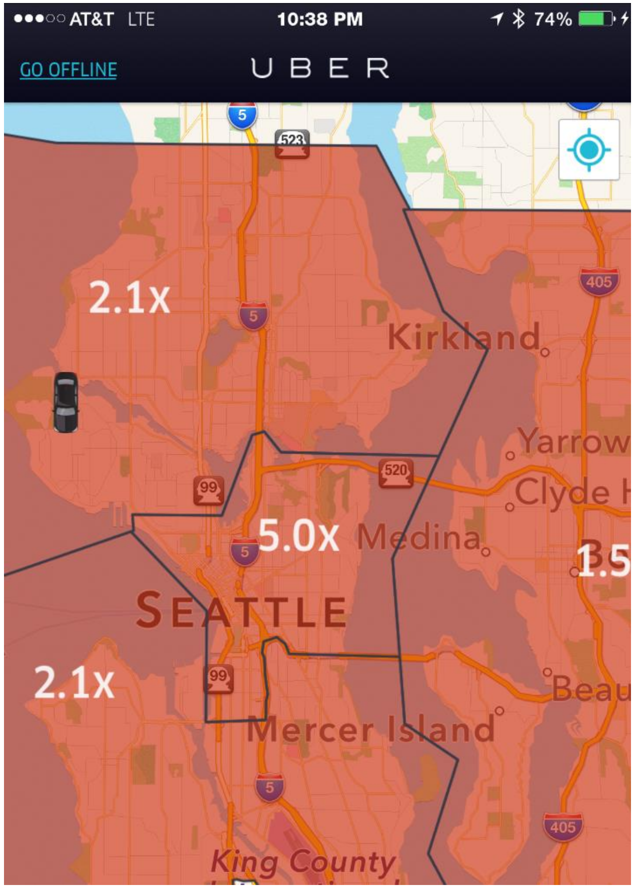
- Surge price interface

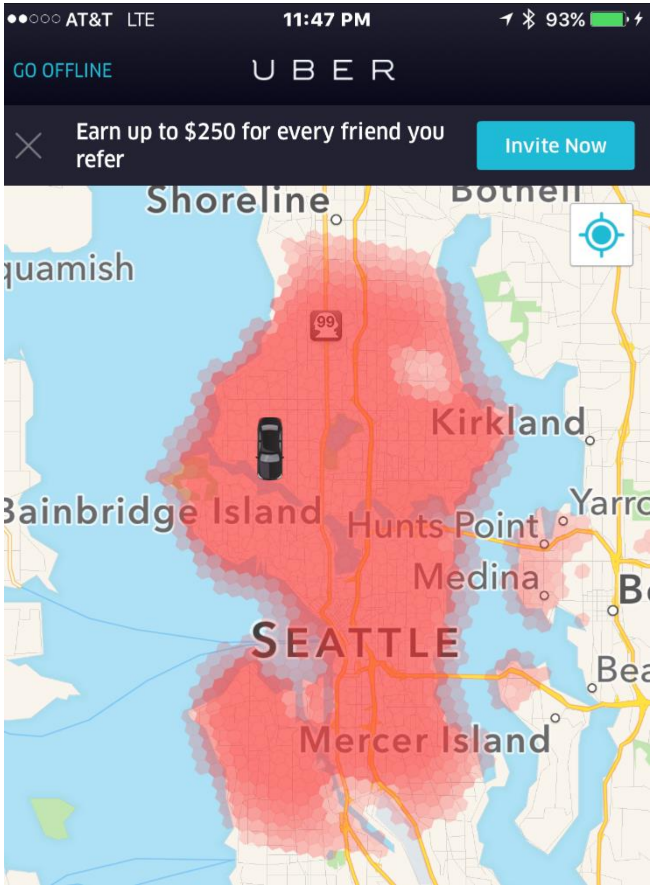




Evolution of Surge Pricing

- Coarse → Fine grained in space

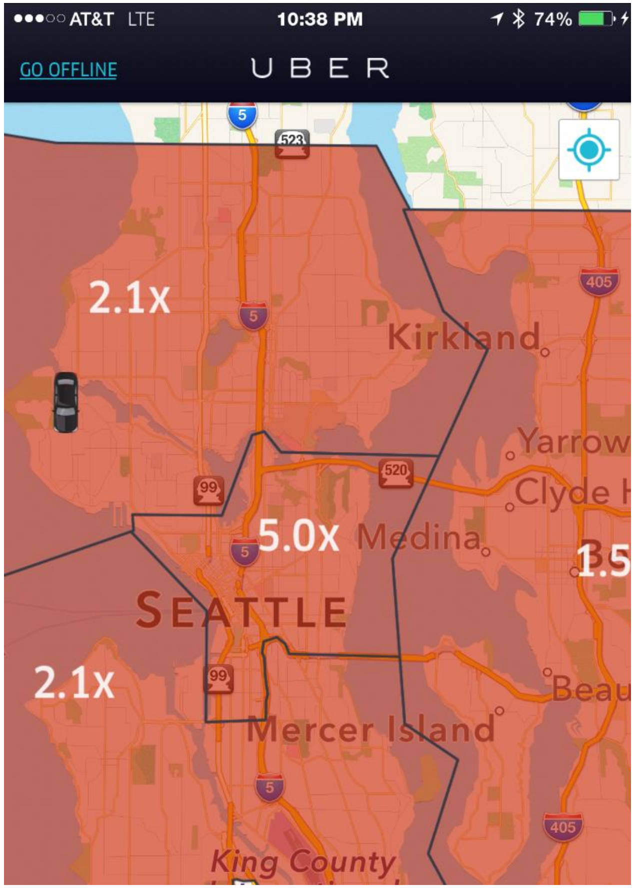




Quiz

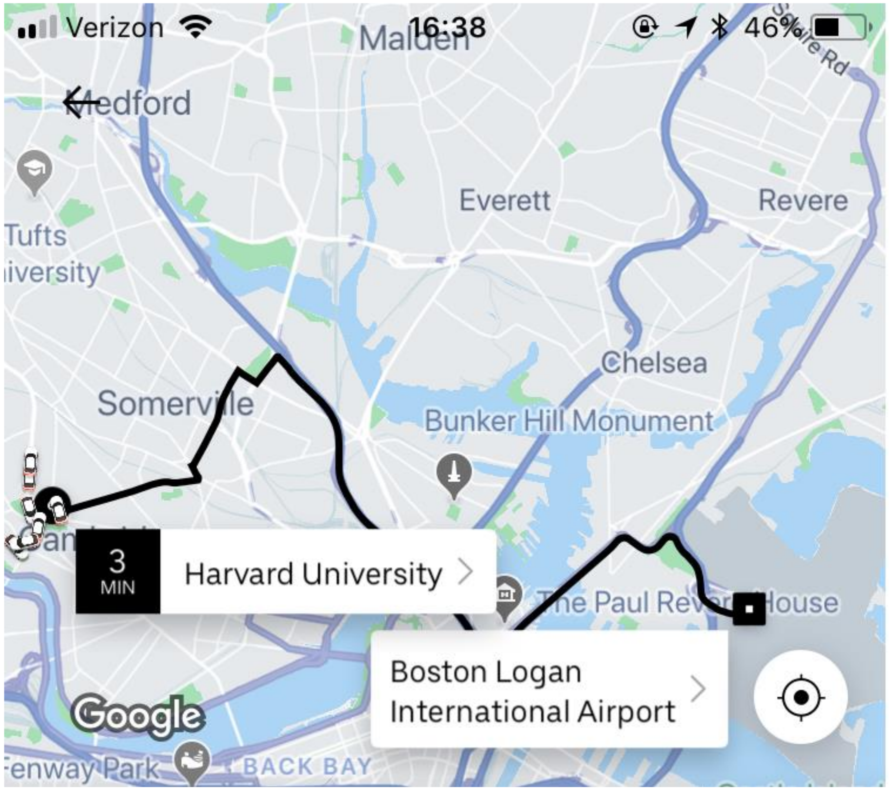
- What are the potential strategic behavior of a driver (with old or new interface)?

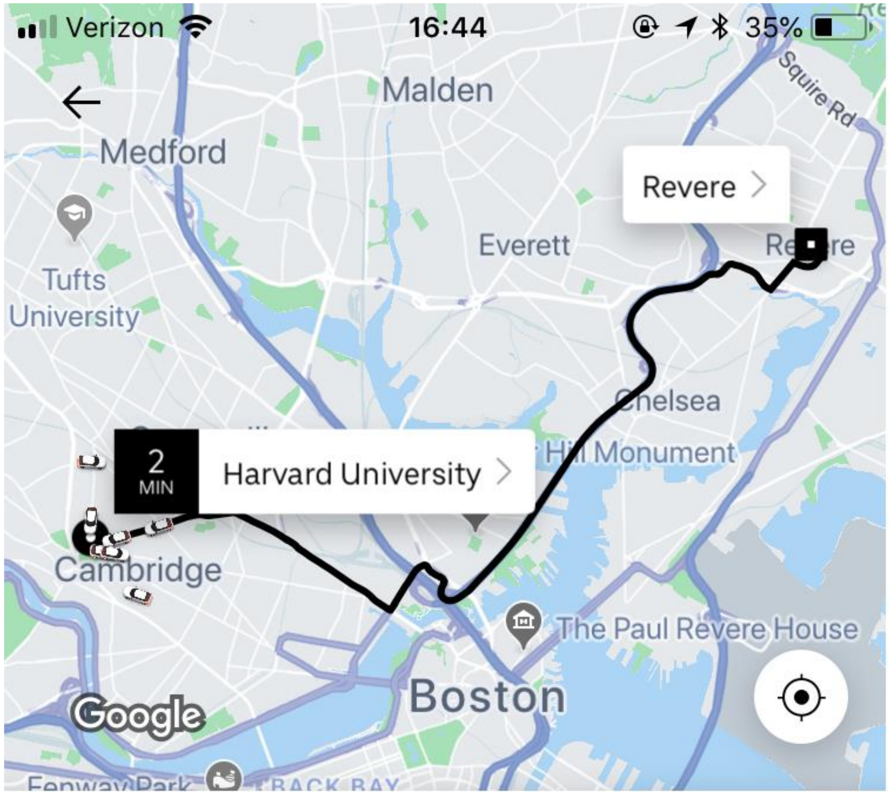
Market Failure - 1



Market Failure - 2

Market Failure - 3





Bad draw dispatches: "after accepting, drivers are able to contact the rider. Some may [] learn [the] destination [] and canceling if [] the trip will not be worth the time."

Competitive Equilibrium

- Competitive Equilibrium (CE)
  - Also called Walrasian equilibrium
  - Traditional concept in economics
  - Commodity markets with flexible prices and many traders

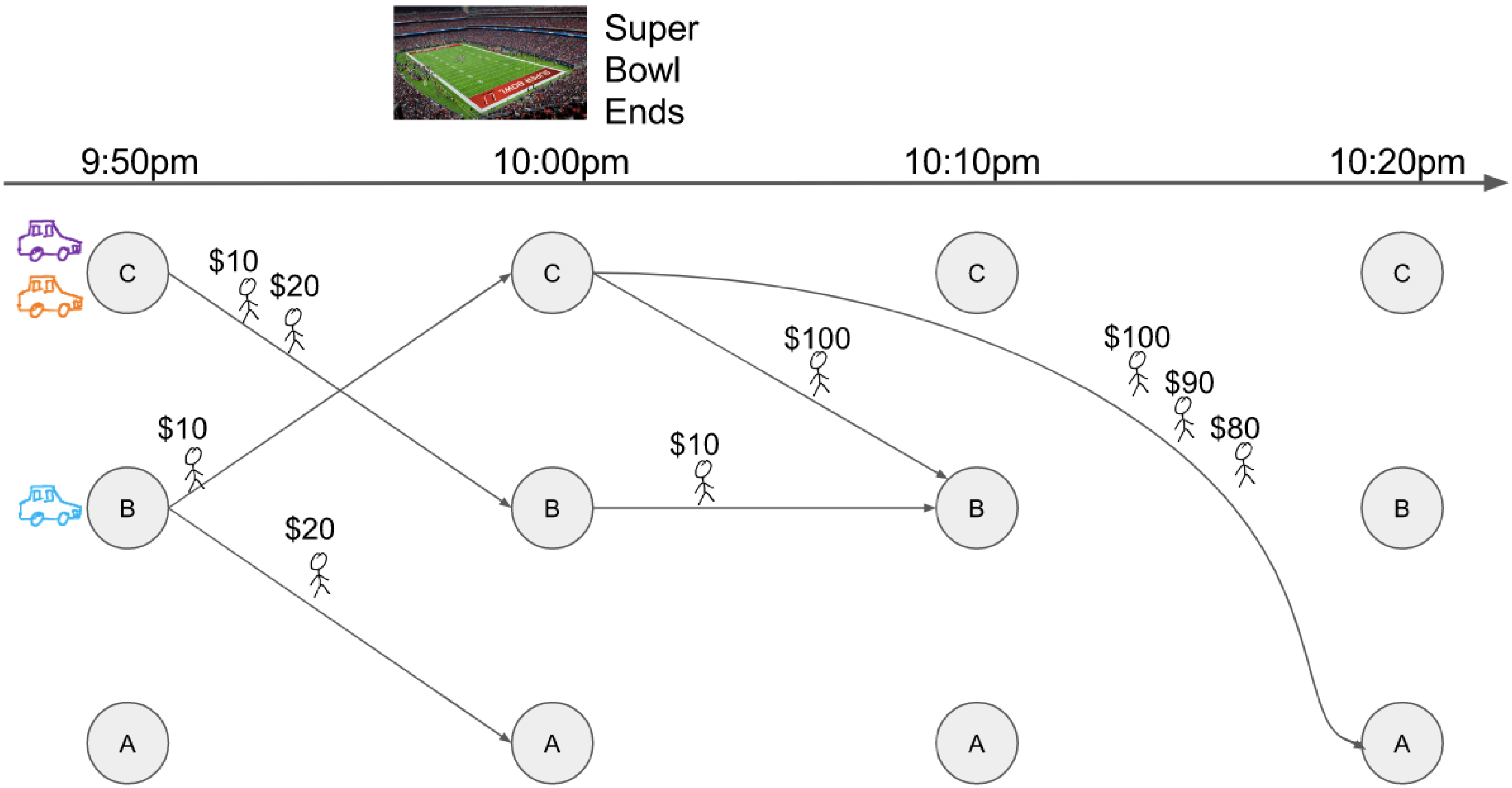
Competitive Equilibrium

- A very simple setting
  - A set of items $[n] = \{1,2, \dots n\}$
  - A set of buyers $[m] = \{1,2, \dots, m\}$
  - Each buyer i has a valuation for each item j: v_{ij}
  - Given a price vector $p \in \mathbb{R}^n$, agent i's utility is: $u_i(x;p) = v_i \cdot x - p \cdot x$ where $x \in \{0,1\}^n$ indicates which items the agent gets
  - Each agent can get at most one item

Competitive Equilibrium

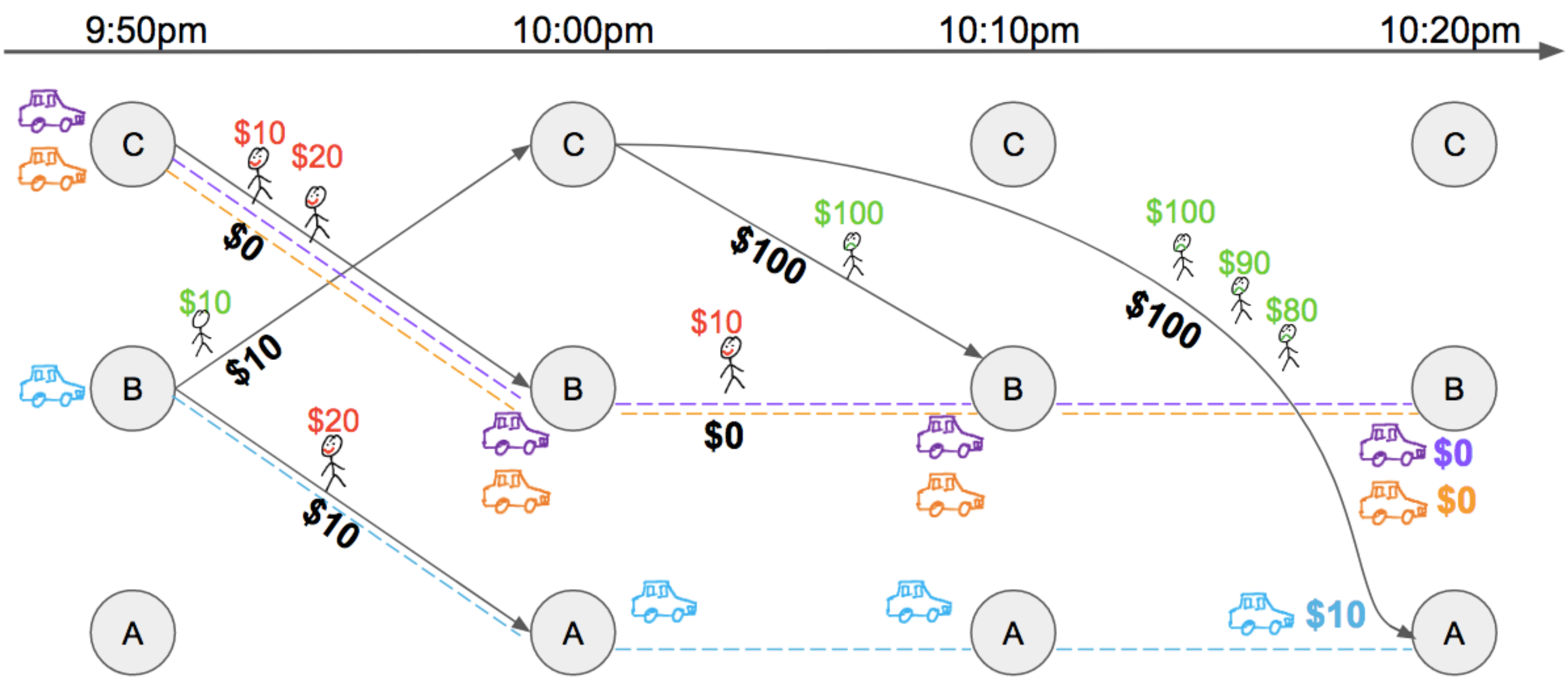
- A CE consists of:
  - A price vector $p \in \mathbb{R}_+^n$
  - A valid allocation matrix x
    - $x_{ij} \in \{0,1\}$ indicates whether or not item j is allocated to agent i
    - Each item is allocated at most once $\sum_i x_{ij} \le 1, \forall j$
    - Each buyer can get at most one item $\sum_j x_{ij} \le 1, \forall i$
    - Use x_i to denote the binary vector for agent i
  - p and x satisfy the following constraints
    - Best response
      - $x_i \in \underset{x: x \in \{0,1\}^n, \sum_j x_j \le 1}{\operatorname{argmax}} u_i(x; p), \forall i$
    - Market clearance
      - $\forall j, \sum_i x_{ij} = 1$ or $p_j = 0$

Super Bowl Example



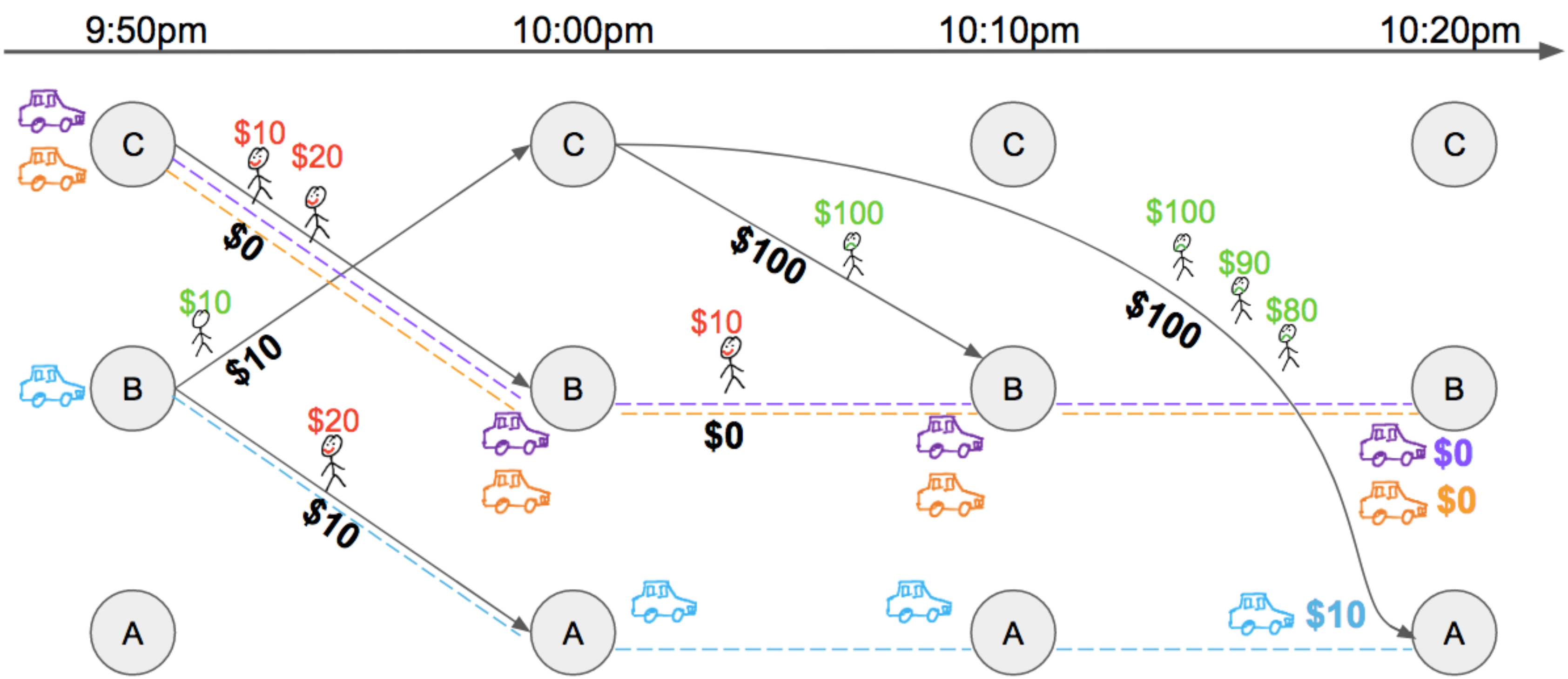
Myopic Pricing

- At current time t, each location has a sub-market
- Allocate cars to the riders with highest valuations
- Driver-pessimal price shown in black



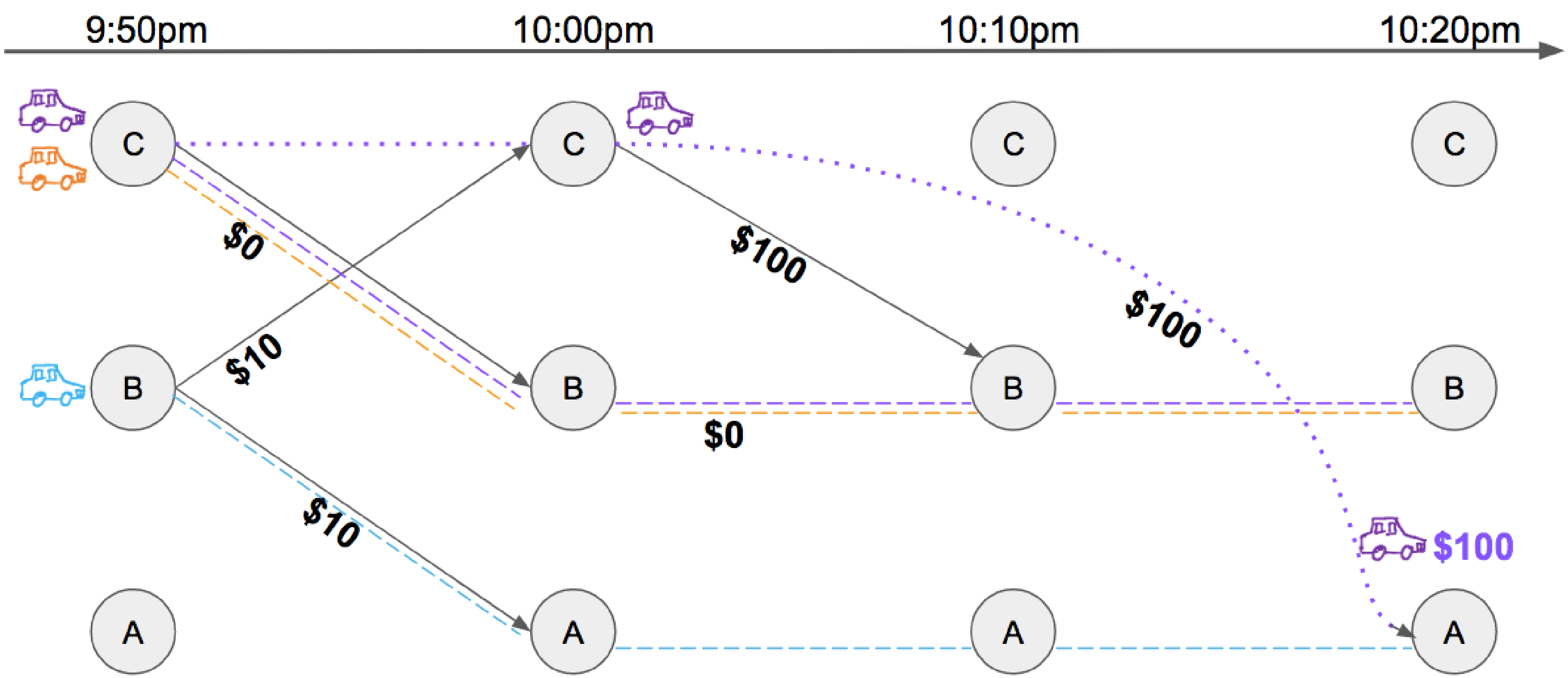
Quiz

- With Myopic Pricing, at most, how much more can the purple driver earn if he deviates from the system's assignment and all other drivers always follow the system's assignment? (Options: $100, $90, $80, $0)



Useful Deviation

- Purple driver rejects the assigned ride at 9:50am to earn more money



Spatial-Temporal Pricing

- Model: Discrete time/location, Impatient riders, Anonymous origin-destination trip price
- One-shot assignment
  - Assignment plan: Decompose a min-cost flow
  - Pricing: Dual of flow LP
  - Form competitive equilibrium (CE)
    - Welfare optimal
    - Maximize total payment for each driver
    - Maximize utility for each rider
    - Envy free
    - All feasible driver payments in CE form a lattice

Spatio-Temporal Pricing for Ridesharing Platforms. Hongyao Ma, Fei Fang, David C. Parkes. In EC-19

ILP for Computing Optimal Assignment Plan

$$\max_{x,y} \sum_{j\in\mathcal{R}} x_j v_j - \sum_{i\in\mathcal{D}} \sum_{k=0}^{|\mathcal{Z}_i|} y_{i,k}\lambda_{i,k}$$

Dual Variables

$$\text{s.t.} \sum_{j\in\mathcal{R}} x_j \mathbb{1}\{(o_j, d_j, \tau_j) = (a,b,t)\} \le \sum_{i\in\mathcal{D}} \sum_{k=0}^{|\mathcal{Z}_i|} y_{i,k} \mathbb{1}\{(a,b,t) \in Z_{i,k}\}, \quad p_{a,b,t} \qquad \forall (a,b,t) \in \mathcal{T}$$

$$\sum_{k=0}^{|\mathcal{Z}_i|} y_{i,k} = 1, \quad \pi_i \qquad \forall i \in \mathcal{D}$$

LP Relaxation

$$\cancel{x_j \in \{0,1\},} \qquad x_j \le 1 \quad u_j \qquad \forall j \in \mathcal{R}$$

$$\cancel{y_{i,k} \in \{0,1\},} \qquad \forall i \in \mathcal{D},\ k = 1, \ldots, |\mathcal{Z}_i|$$

$$x_j \ge 0$$

$$y_{i,k} \ge 0$$

Dual Problem to Compute CE Pricing

$$
\begin{aligned}
\min \quad & \sum_{i \in \mathcal{D}} \pi_i + \sum_{j \in \mathcal{R}} u_j \\
\text{s.t. } \pi_i \geq & \sum_{(a,b,t) \in Z_{i,k}} p_{a,b,t} - \lambda_{i,k} && \forall k = 0, 1, \ldots, |\mathcal{Z}_i|, \ \forall i \in \mathcal{D} \\
& u_j \geq v_j - p_{o_j, d_j, \tau_j}, && \forall j \in \mathcal{R} \\
& p_{a,b,t} \geq 0, && \forall (a, b, t) \in \mathcal{T} \\
& u_j \geq 0, && \forall j \in \mathcal{R}
\end{aligned}
$$

Spatial-Temporal Pricing

- However…Drivers can deviate and trigger recomputation!

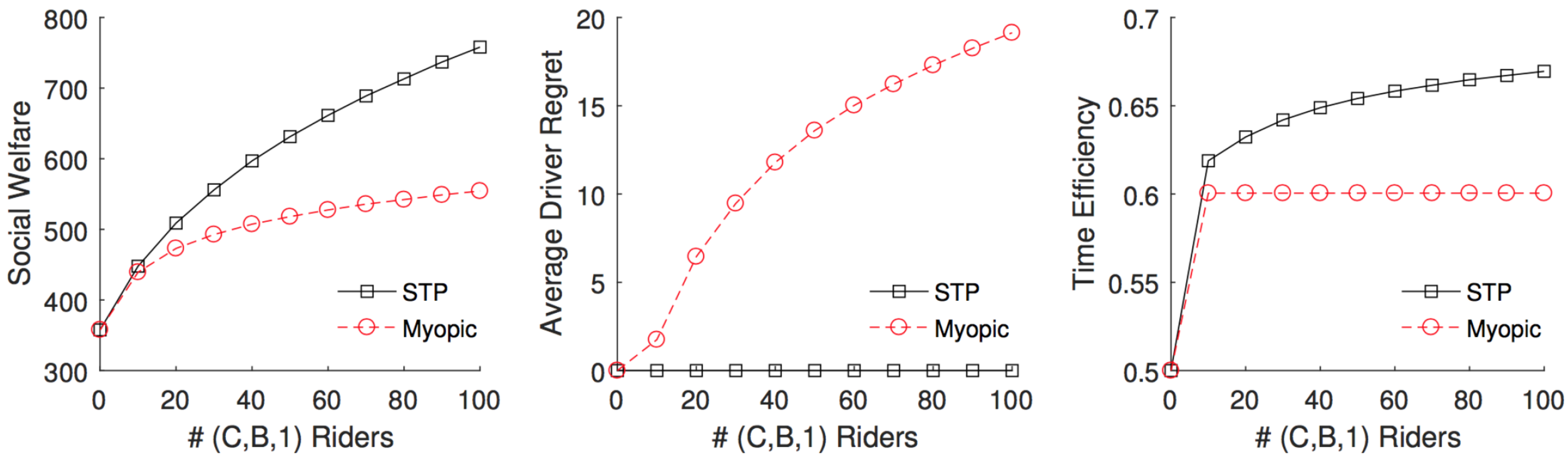
- Solution: Driver-Pessimal CE
  - Trip price = welfare gain difference

$$p_{a,b,t} = \Phi_{a,t} - \Phi_{b,t+dist(a,b)}$$
$$\Phi_{a,t} \triangleq W(D \cup \{(t,T,a)\}, R) - W(D,R)$$

  - Incentive compatible subgame perfect equilibrium
  - No driver want to deviate from assigned action!

Spatial-Temporal Pricing

- SPT vs Naïve surge



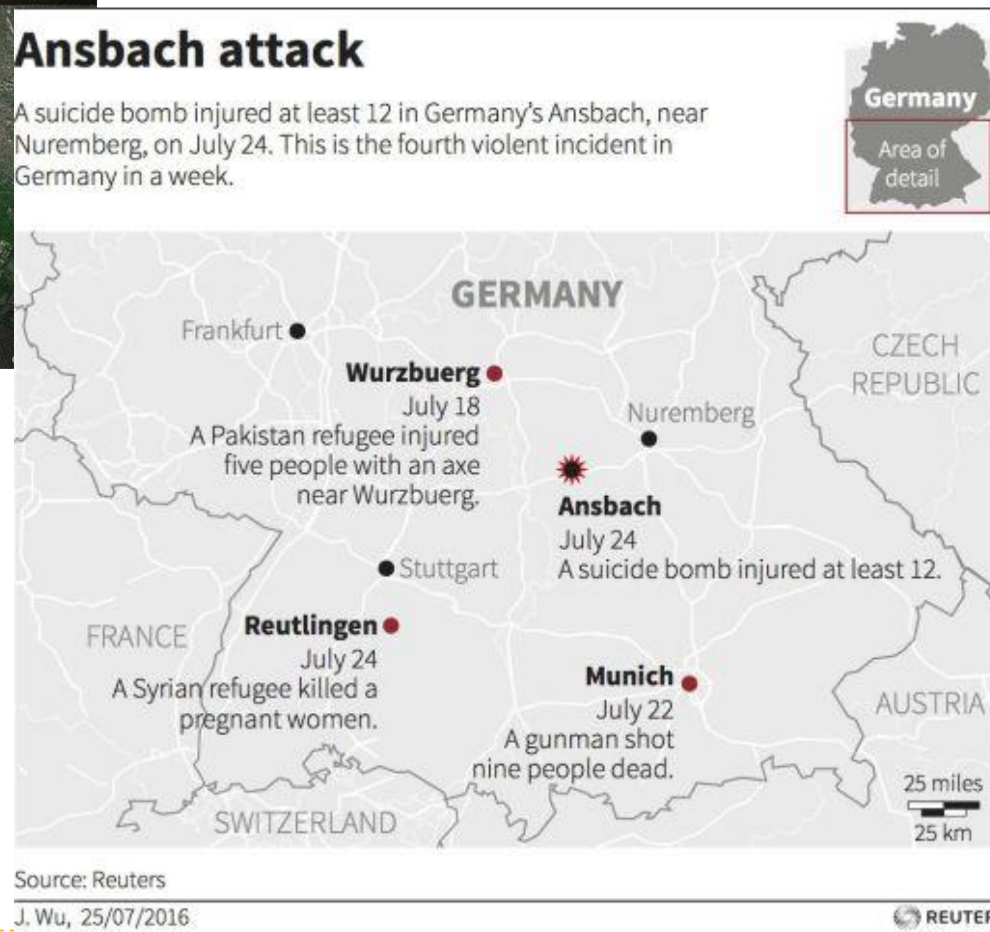
Summary

- Games with Human Players for Real-world Applications
- End-to-End Learning and Decision Making in Games
- Learning-Powered Strategy Computation in Large Games

Thank you!

Fei Fang
Carnegie Mellon University
feifang@cmu.edu

Security Challenges



Sustainability Challenges

Mobility Challenges