Deep Learning for Solving Large Scale Complex Games

Bo AN

boan@ntu.edu.sg

School of Computer Science and Engineering

Nanyang Technological University

June 23, 2022

AMI *Agent Mediated Intelligence Research Group*

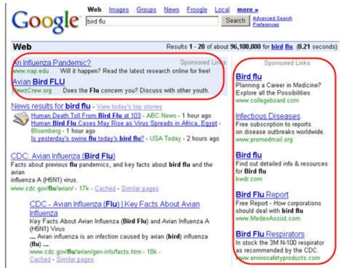
Many AI Problems are Games

- AI: study and construction of *rational* agents *[Russell & Norvig, 2003]*

building a single agent (1950s-70s) →	*Multi-agent systems (cooperative)* (1980s-) →	*Multi-agent systems (competitive)* (1995-) →

- Many real world problems are games and game theory is needed (1940s-)

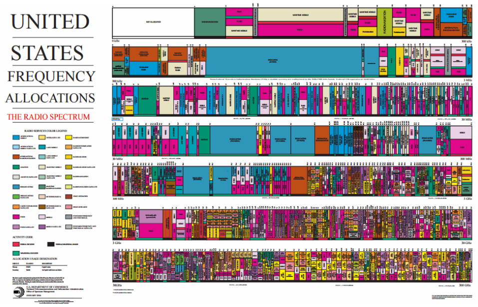




- GT for AI: success in computer poker, security, auction…

Google™

bing



- The rest of the talk:
  - Solving games with algorithmic game theory
  - New trend: Deep (reinforcement) learning for solving games

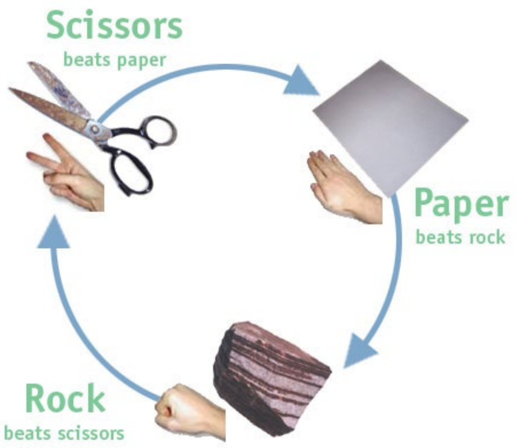
Games and Computation

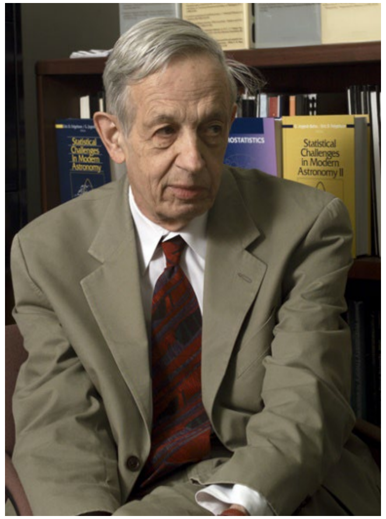
- Players, strategies, payoffs

Player B

Player A

0, 0	1,-1	-1,1
-1, 1	0, 0	1, -1
1, -1	-1, 1	0, 0





- Nash Equilibrium: no agent has incentive to unilaterally deviate
  - In any (finite) game, at least one Nash equilibrium (possibly mixed) exists *[Nash, 50]*
  - In 2-player zero-sum games, a profile is an NE iff both players play minimax strategies
  - Computing one (any) Nash equilibrium is PPAD-complete (even in 2-player games) *[Daskalakis, Goldberg, Papadimitriou 2006; Chen, Deng 2006]*
  - All known algorithms require exponential time (in the worst case)
    - Lemke-Howson, support enumeration
- Mechanism design

Libratus for Computer Poker [Sandholm@CMU]



Abstraction (offline)

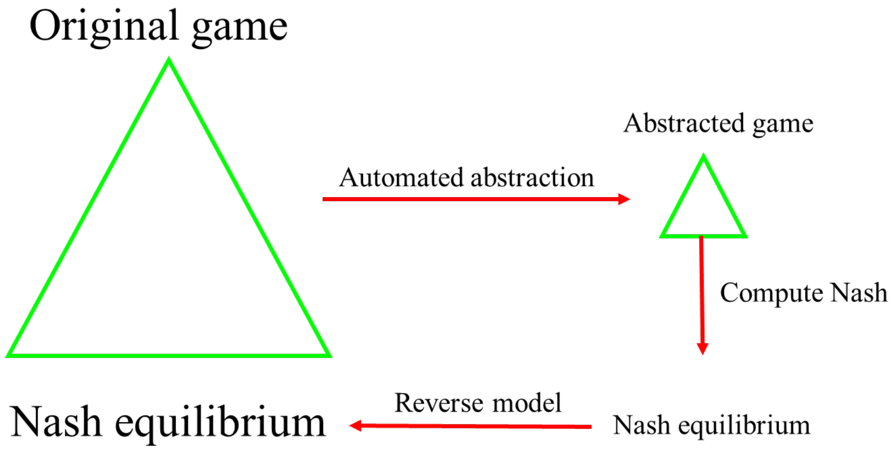
- Action abstraction
- Card abstraction
- Game size from 10^{161} to 10^{12}

Equilibrium Finding (offline)

- CFR
- CFR^{+}
- Monte Carlo CFR

Decomposition and Subgame Refinement (online)

- endgame solving
- subgame re-solving
- max-margin subgame refinement



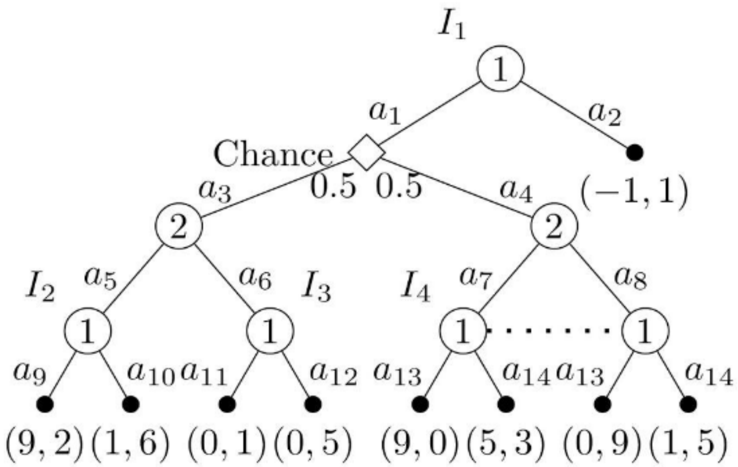
➢ Deep learning: Alberta's DeepStack, DeepMind

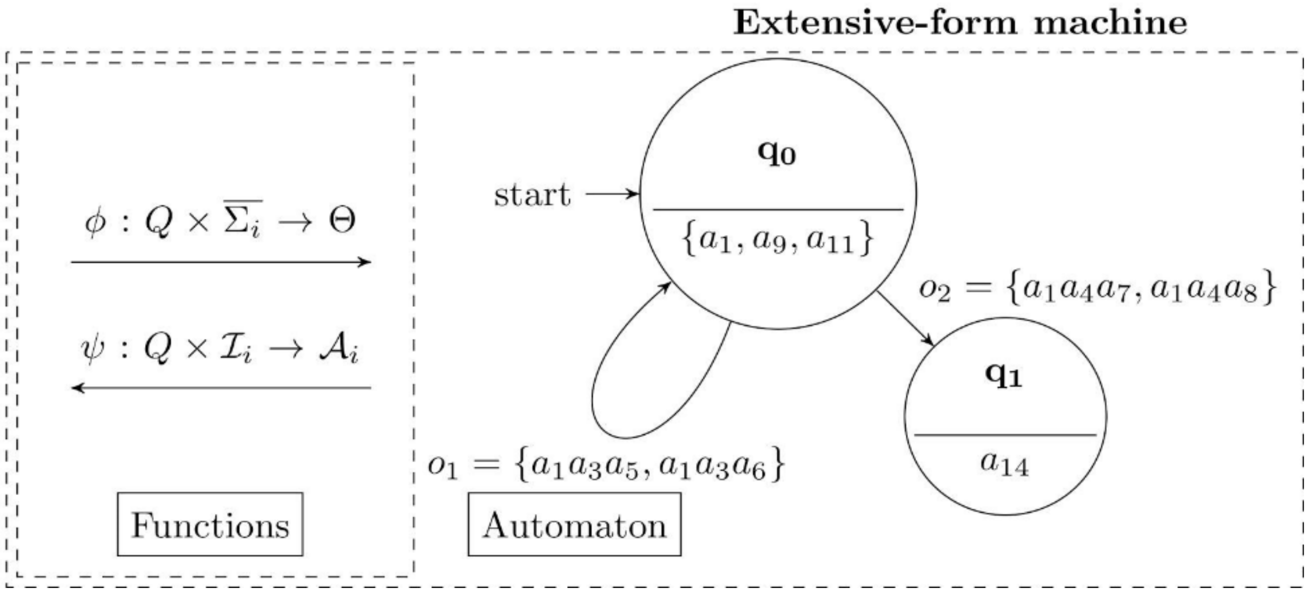
Playing Games with Machines [EC'20]

- Strategies differ in their implementation complexity
- If we knew how to model complexity → **Machines**
  - We can focus on **relevant** (easy) strategies

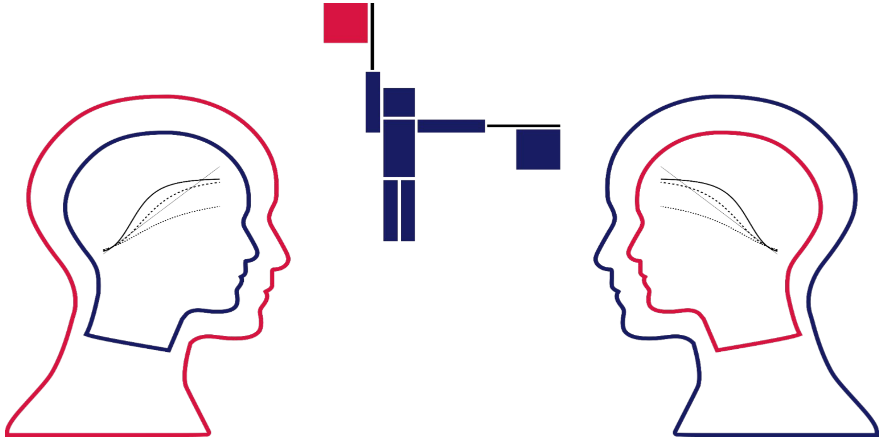
COROLLARY 4.4. *Let $\mathcal{L}$ be a size-parametric class of perfect-recall EFGs with 2 players and $\mathcal{M}_f^S(n)$ be a small class of machine strategies of the follower in $\mathcal{L}(n)$. Then the problem of finding a strategy profile $\gamma^{SSE} = (\gamma_l^R, M_f)$ describing an SSE in a restriction of $\mathcal{L}(n)$ induced by $\mathcal{M}_f^S(n)$, i.e., $M_f \in \mathcal{M}_f^S(n)$, is polynomial.*

- Lower **computational & implementation**





Computing Quantal Correlated Equilibrium in NFGs [EC'22]



- Correlated equilibrium = coordination
  - *Signaller sends signals to players*
  - *Assumption: everyone is rational*
- Real-world players are subrational!
- Our contributions:
  - *Incorporating quantal-response behavior:*

$$u_i(\delta_{-i}, a_i^k) \leq u_i(\delta_{-i}, a_i^l) \Rightarrow QR_i^k(\delta_{-i}) \leq QR_i^l(\delta_{-i}) \qquad \forall \delta_{-i} \in \Delta_{-i}, a_i^k, a_i^l \in A_i,$$

  - *Analyzing quantal correlation: relations to other equilibria, topology, complexity*

Theorem 1. *Let $G = (N, A, S, u)$ be a signaling game. Then (i) every quantal response equilibrium in G is a quantal correlated equilibrium with trivial signaling structure; (ii) the limit of quantal correlated equilibria in G is a correlated equilibrium in G as quantal responses approach the best response; (iii) the signaling correspondence $\lambda \to QCE(\lambda)$ is upper hemicontinuous; and (iv) computing a quantal correlated equilibrium in G is PPAD-hard.*

  - *Formulating computational methods for*
    - ***tracing*** quantal correlated ***equilibrium*** using a homotopic system
    - ***gradiently*** optimizing the ***signal*** structure

Game Theory for Security [60+@ AAAI, AAMAS, IJCAI, NeurIPS, ICML]

- Global challenges for security


Boston Marathon bombings


French oil tanker hit by a boat


Cyber physical attacks

- Security resource allocation
  - ❑ Limited security resources
  - ❑ Adversary monitors defenses, exploits patterns
- We pioneered the first set of applications of game theory for security resource scheduling(2007-)

- ❑ 60+ papers at premier conferences/journals, 2 best paper awards
- ❑ INFORMS Daniel H. Wagner Prize for Excellence in Operations Research Practice (2012), etc
- ❑ Operational Excellence Award from US Coast Guard (2012), etc
- ❑ Media reports: FOX News, CNN News, Federal News Radio, Defense News, The Economics Times, Los Angeles Times, etc
- ❑ United States congressional hearing (4 times)

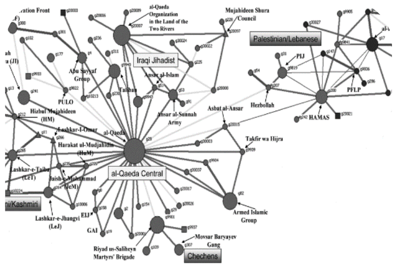






Analyzing Complex Security Games [2016-]

Dynamic Payoff



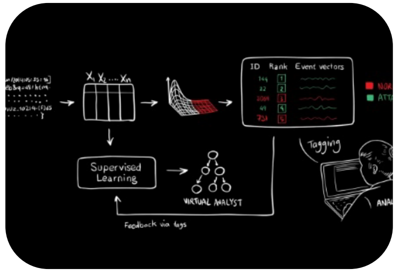
Network Games

Protection Externality

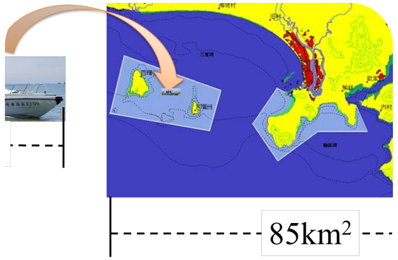
Uncertainty

Strategic Deception

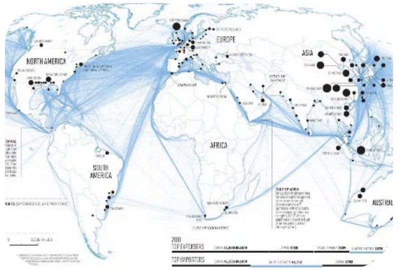
Cyber Security



Adversarial Machine Learning



Coral Reef



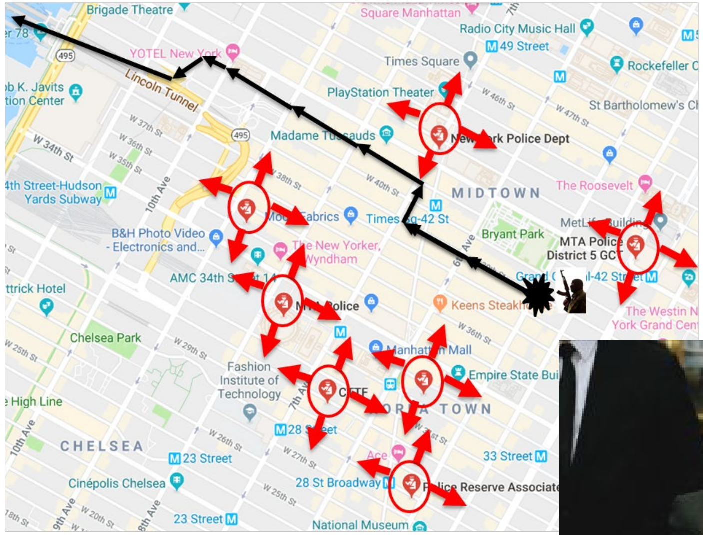
Nuclear Smuggling

Elections

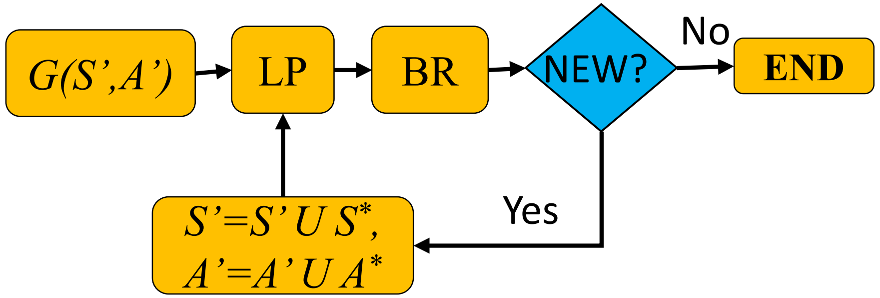
- Combining techniques from AI, Game Theory, Operations Research …
- Marry theory with practice
- Approaches can be applied to other domains *With proper tuning & extension*
  - ❑ Incremental strategy generation
  - ❑ Construct (multiple) equivalent games
  - ❑ Exploit compact representation
  - ❑ Abstraction
  - ❑ Tradeoff between optimality and efficiency
  - ❑ Approximation

Converging to Team-Maxmin Equilibria [ICML'20, AAAI'21, IJCAI'22]

- Equilibria in Multiplayer Games
  - *Hard to compute: PPAD-Complete*
  - *Hard to select: NEs are not unique*
  - *Few results:*
    - Special structure: congestion games
    - No theoretical guarantee: Pluribus *[Brown and Sandholm 2019]*
- Team-Maxmin Equilibria *[von Stengel and Koller 1997]*
  - *A team of players independently plays against an adversary*
  - *Unique in general*
  - *FNP-hard to compute a team-maxmin equilibrium*
    - Formulated as a non-convex program
    - Solved by a global optimization solver
- Converging to Team-Maxmin Equilibria
  - *Existing ISG for multiplayer games*
    - Converge to an NE but many not to a TME
    - Difficult to extend the current ISG to converge to a TME
  - *ISGT: the first ISG guaranteeing to converging to a TME*
    - Conditions in ISGT cannot be further relaxed
  - *CISGT: further improve the scalability*
    - Initialize the strategy space by computing an equilibrium that is easier to be computed
- Unsatisfactory scalability!



Incremental Strategy Generation (ISG)



L×W	5×5	5×5	5×5	5×5	5×5	4×4	6×6	8×8	10×10
(p,q)	(0.8,0.6)	(0.7,0.5)	(0.6,0.4)	(0.5,0.3)	(0.4,0.2)	(0.4,0.2)	(0.4,0.2)	(0.4,0.2)	(0.4,0.2)
FullTME		∞	448s	50.4s	17.8s	0.3s	∞		
ISGT					>1000s	4s	>1000s		
CISGT	9.8s	5.9s	4.7s	3.7s	2.3s	2.2s	8.3s	24s	57s

Table 1. Computing TMEs: ∞ represents out of memory.

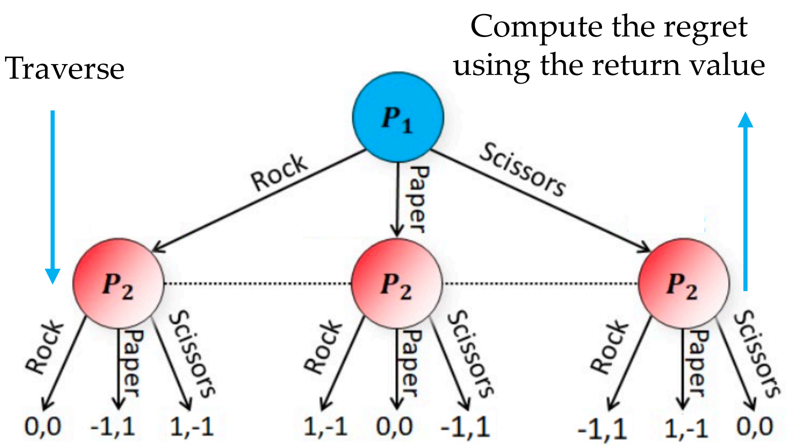
When do We Need D(R)L for Complex Games?

- When GT is better than ML
  - *Requires no data*
  - *No assumption about players' behavior*
  - *Not exploitable*
  - *Theoretical guarantee*
- ML might be more appropriate when
  - *Large scale: Millions of (even continuous) pure strategies*
  - *Uncertainty*
  - *Cannot be well modelled*
  - *Non-convex and cannot be approximated*
  - *No domain structures can be exploited*
- D(R)L for games is receiving increasing attention
  - *Solving games, e.g., DeepStack*
  - *Mechanism design*
- Does not mean D(R)L can always work!
- Rest of the talk: quick overview of our two works since 2021
  - *CFR-MIX: based on counterfactual regret minimization*
  - *NSG-NFSP: based on fictitious play*
  - *NSGZero: based on neural Monte Carlo tree search*

Counterfactual Regret Minimization

- CFR *[Zinkevich et al. 2008]*
  - *A popular algorithm to solve imperfect-information extensive-form games*
  - *In every iteration, it traverses the whole game tree and computes counterfactual regret value for every information set*

$$v_i(I,\sigma) = \sum_{z \in Z_I} \pi_{-i}^{\sigma}(z[I])\pi^{\sigma}(z[I],z)u_i(z) \qquad r^t(I,a) = v_i(I,\sigma_{I\to a}^t) - v_i(I,\sigma^t)$$



Example: Rock-paper-scissors game. There are only one information set for each player.

  - *Compute the strategy of next iteration using regret matching based on the sum of counterfactual regret values*

$$R^T(I,a) = \sum_{t=1}^{T} r^t(I,a) \qquad R^{T,+}(I,a) = \max\{R^T(I,a),0\}$$

$$\sigma_i^{T+1}(I)(a) = \begin{cases} \dfrac{R_i^{T,+}(I,a)}{\sum_{a \in A(I)} R_i^{T,+}(I,a)} & if \displaystyle\sum_{a \in A(I)} R_i^{T,+}(I,a) > 0 \\ \dfrac{1}{|A(I)|} & otherwise \end{cases}$$

  - *The average strategy over all iterations converges to Nash equilibrium in two-player zero-sum games*

- CFR variants
  - *Sampling-based CFR*
    - Traversing the whole game tree is very time-consuming. Sampling-based CFR only traverses a subset of the game tree.
      - External Sampling CFR, Outcome Sampling CFR *[Lanctot et al., 2009]*, Probe sampling*[Gibson et al., 2012]*
  - *Deep-based CFR*
    - Tabular representation needs huge memory. Deep-based CFR uses neural networks to represent the regret value and strategy.
      - Deep CFR *[Brown et al., 2019]*, Double Neural CFR *[Li et al., 2019]*

CFR-MIX: Solving Imperfect Information Extensive-Form Games with Combinatorial Action Space [IJCAI'21]

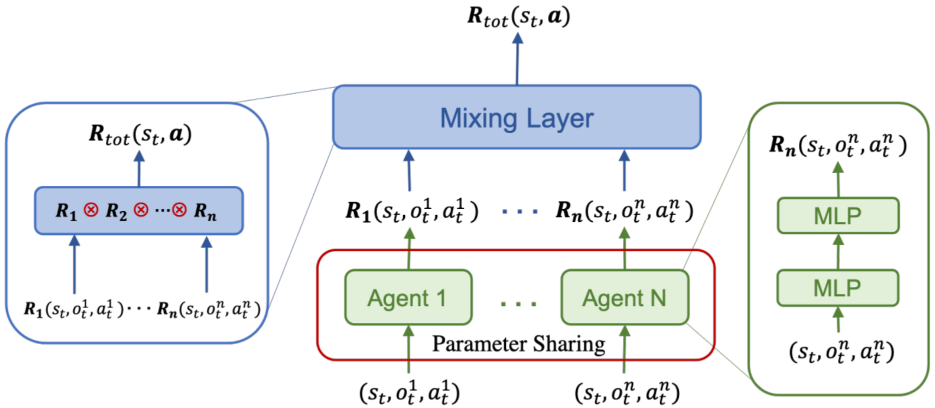
➢ Team-Adversary Games

- ❑ *A team of players cooperatively plays against one adversary*
- ❑ *Hard to solve due to the large combinatorial action space*
  - ❖ The exponentially growing joint action space of the team
- ❑ *Ineffectiveness of existing methods*
  - ❖ Out of memory for tabular-form methods
  - ❖ Ineffective to train the strategy network over the large action space for DNNs

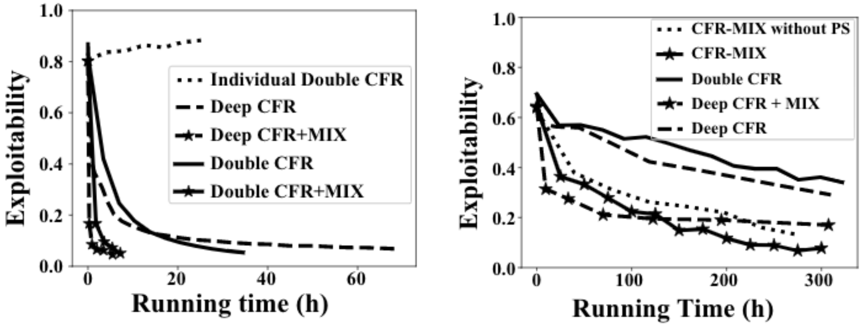
➢ CFR-MIX

- ❑ *Use the individual strategy representation to reduce the strategy space*
  - ❖ $f_{\mathbb{T}} = (\sigma_1, \sigma_2, \ldots, \sigma_n)$ where σ_i is the strategy for team player i
- ❑ *Provide the **consistency** relationship to maintain the the NE unchanged*
  - ❖ $\sigma_{\mathbb{T}}(I, a) = \sigma_1(I, a_1)\sigma_2(I, a_2) \ldots \sigma_n(I, a_n)$ where $\sigma_{\mathbb{T}}$ is the joint strategy for the team
- ❑ *Propose a Product-form decomposition method to maintain the consistency*
  - ❖ $\forall\, a,\ \ R_{tot}(I, a) = \prod_{i=1}^{n} R_i(I, a_i)$
- ❑ *Implement the decomposition method using a mixing layer*
- ❑ *Theorem: With the probability $1 - \rho$, the total regret of player i at time T is bounded by*

$$R_i^T \leq \left(1 + \frac{\sqrt{2}}{\sqrt{\rho K}}\right) \Delta\, |I_i| \sqrt{|A|}\sqrt{T} + 4T|I_i|\sqrt{|A|\Delta_{\epsilon_L}}$$



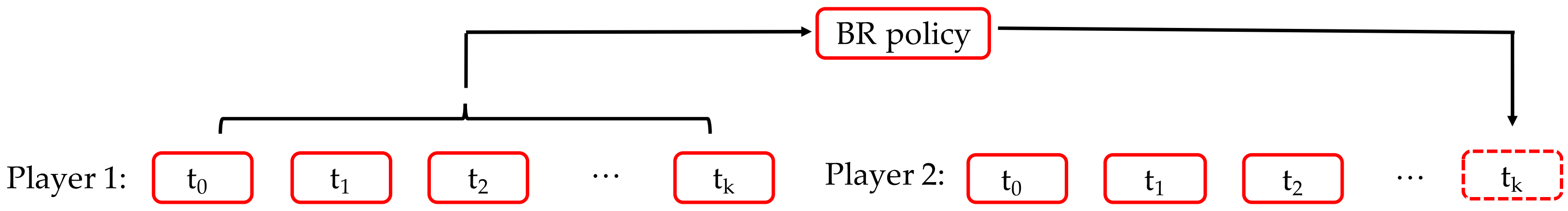
Architecture of team's regret neural network



(a) 3*3 grid, 1 vs 2
(b) 5*5 grid, 1 vs 4

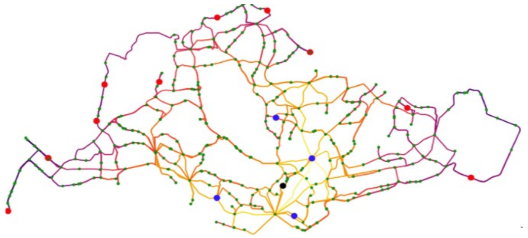
Neural Fictitious Self-Play

- Fictitious play is a game-theoretic algorithm for learning NE
  - Players repeatedly play a game. At each iteration, each player plays with its opponent's past average policy and best responds against it

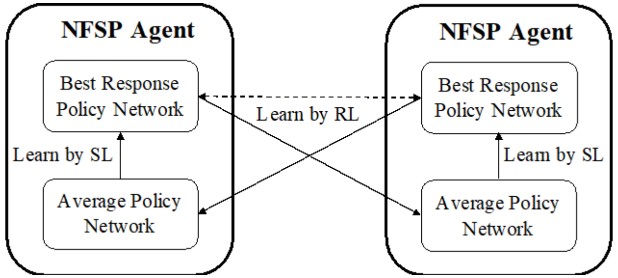


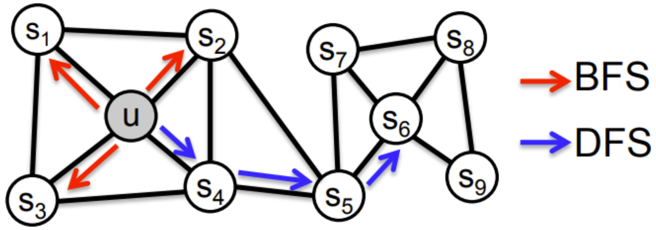
  - Full-Width Extensive-Form Fictitious Play (XFP) *[Heinrich et al. 2015]* extends fictitious play from normal form to extensive form
- Neural Fictitious Self-Play *[Heinrich et al. 2016]*
  - A sampling and machine learning-based adaption of XFP
  - Utilizing deep neural network function approximation
  - Each agent consists of two neural networks, i.e., the average policy network and the best response (BR) policy network
  - The average policy network approximates an agent's past average policy by supervised learning
  - The BR policy network best responds to other agents' average policy by reinforcement learning
- Related work
  - OptGradFP *[Kamra et al. 2018]* firstly introduces fictitious play to continuous action spaces
  - NFSP cannot solve games like NSGs whose action space is extremely large

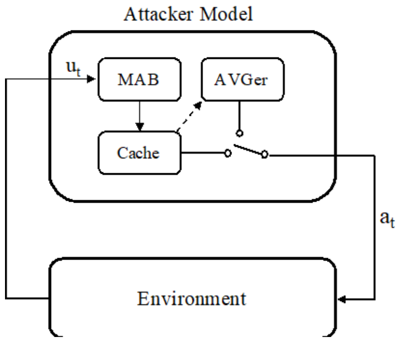
Solving Large-Scale Extensive-Form Network Security Games via Neural Fictitious Self-Play [IJCAI'21]

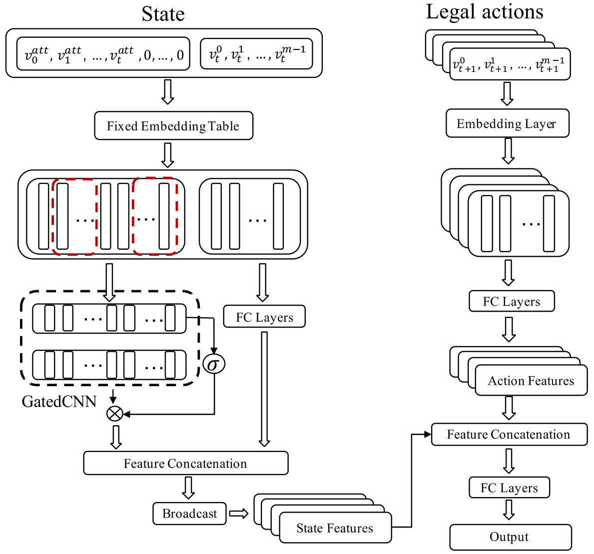


- Securing networked infrastructure with limited security resources
  - Vanilla NFSP cannot solve NSGs because
    - The defender has combinational action space, and its legal action spaces change with states
    - It is impossible for the output of deep neural networks to cover the huge action space
    - The output of deep neural networks have inconsistent semantics due to the changing legal action spaces
  - *Sparse reward which brings difficulties in exploration*
  - *How to represent the road network efficiently when graphs are extremely large*
- NSG-NFSP: approximating an NE defender policy in NSGs
  - *Framework: Neural Fictitious Self-Play (NFSP), which is guaranteed to converge to an NE*
  - *Learning state and action representations when approximating BR and AVG policies*
  - *Enabling NFSP with high-level actions for efficient exploration*
  - *Learning efficient graph node embeddings via node2vec*









NSGZero: Efficiently Solving Large-scale Network Security Games via Neural Monte Carlo Tree Search [AAAI'22]

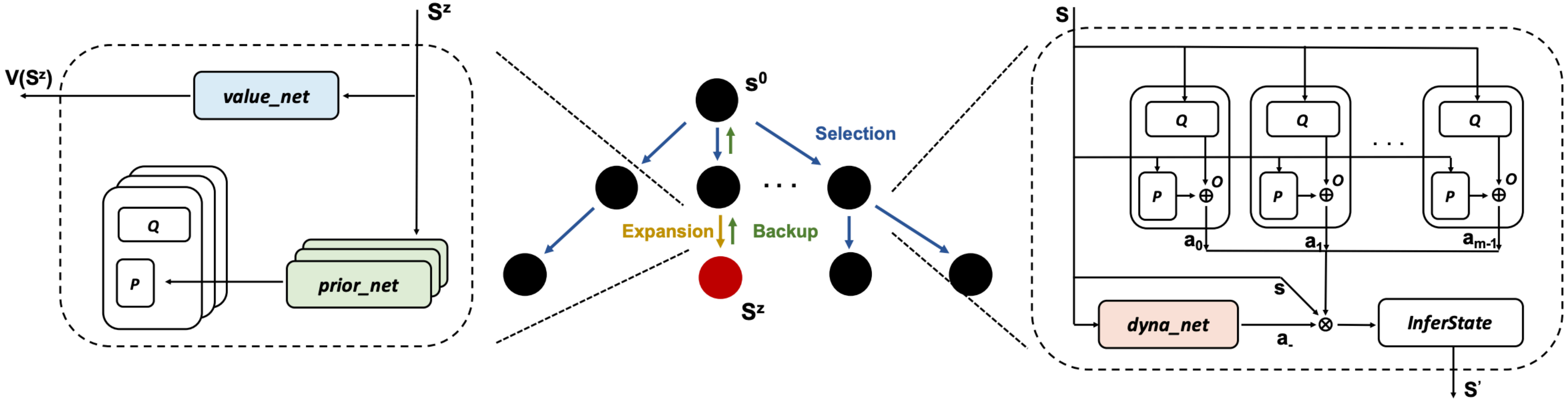
- Improving data efficiency by performing planning with neural MCTS
  - Modeling the dynamics of NSGs
  - Predicting future state values when planning
  - Leveraging prior knowledge to do exploration
- Improving scalability by enabling neural MCTS with decentralized execution
  - Agents record simulation statistics separately

Algorithm 2: NGSZero-EXECUTION

Input: The current state s_t, the search tree Ψ.

$\Psi.clear()$ \\ clear statistics stored in the search tree;
for N *simulations* **do**
  $\Psi.search(s_t)$ \\ perform lookahead search;
end
for *resources* $i = 0, \ldots, m-1$ **do**
  $\pi_i(s_t, a_i) \propto \frac{O(i,s_t,a_i)^{1/\mathcal{T}}}{\sum_{b_i} O(i,s_t,b_i)^{1/\mathcal{T}}}$; $a_{i,t} \sim \pi_i(s_t)$;
end

Output: Joint action $\boldsymbol{a}_t = \langle a_{0,t}, \ldots, a_{m-1,t} \rangle$.

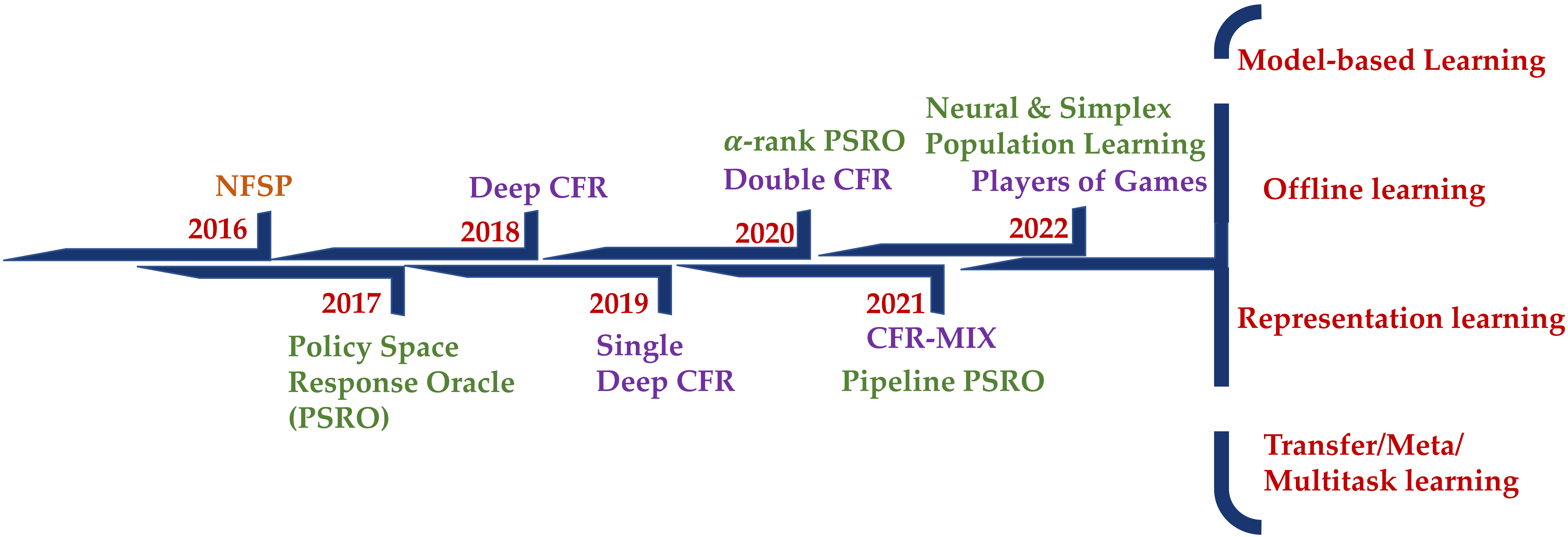


Value network: predict the state value of the next state from backup phase

Prior network: predict the prior policy of resources

Dynamics network: predict the next state

Roadmap and Next Steps on DL for Solving Games



DL + GT: The Future

- Many DL **Scenarios** can be viewed as **Games**
  - Generative Adversarial Networks (GAN)
  - Adversarial training
  - Adversarial reinforcement Learning
  - **Even**, transfer/meta/multitask learning, and self-supervised learning
- GT provides **theoretical** and **algorithmic** tools
  - Theoretical: regret analysis, convergence rate
  - Algorithmic: CFR, NFSP, PSRO
- D(R)L provides **representations** and **training** methods
  - RL for general-sum games?
  - The limit of representation learning **[ICLR'22, ICML'22]**
  - Big models, e.g., transformer
- There is a long way to go…