Surrogate Bayesian Networks for Approximating Evolutionary Games

Vincent Hsiao[1], Dana Nau[1], Rina Dechter[2]

DEPARTMENT OF COMPUTER SCIENCE

University of Maryland, College Park[1], University of California, Irvine[2]

Background

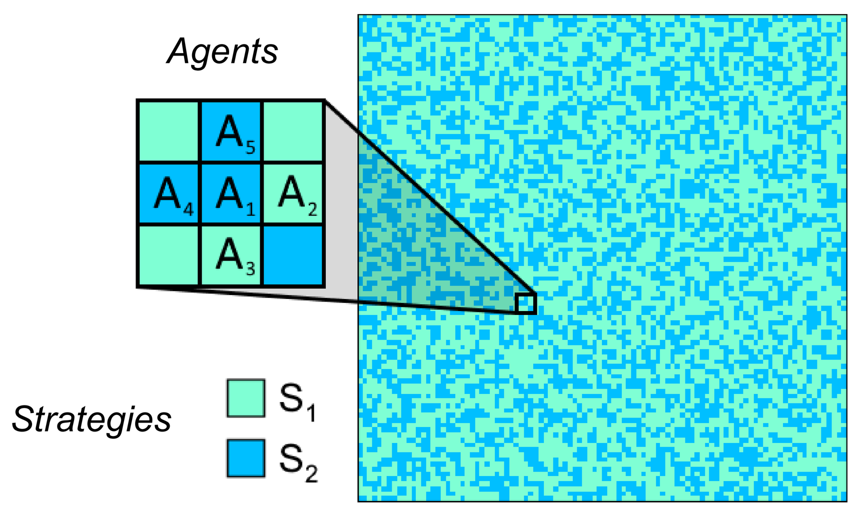
Evolutionary Game Theory (EGT)

- Application of game theory to evolving populations

Spatial Evolutionary Game

- EGT model on structured population (e.g. grid)
- Spatial EGT = (S, U, G, F)
  - S - set of strategies
  - U - payoff matrix/function
  - G - graph of population structure
  - F - replicator rule (e.g. Fermi rule)

$$U = \begin{array}{c|cc} & s_1 & s_2 \\ \hline s_1 & a & b \\ s_2 & c & d \end{array}$$



Goal: evaluate population dynamics over time, e.g. proportion of pairs of adjacent agents playing strategy i and j at a given time T.

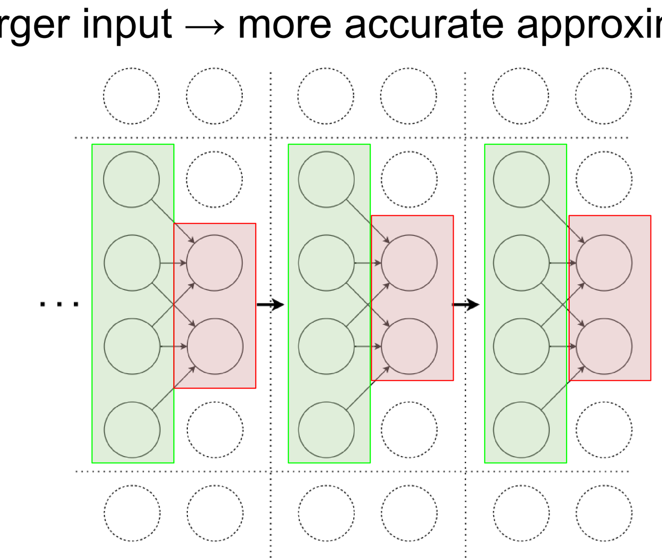
Bayesian Network (BN)

- Bayesian Network: graphical model (X,D,F)
  - Variables: $X = \{X_1, X_2, ..., X_N\}$
  - Domains: $D = \{D_{X_1}, D_{X_2}, ..., D_{X_N}\}$
  - Parent Functions: $F = \{F_1, F_2, ..., F_N\}$

Problem Statement

Current Approach

- Use Symmetric Dynamic Bayesian Network Approximation (SD-BNA) to model spatial EGT dynamics
  - SD-BNA's have input (green) and output (red) sections at each iteration
  - Larger input → more accurate approximation



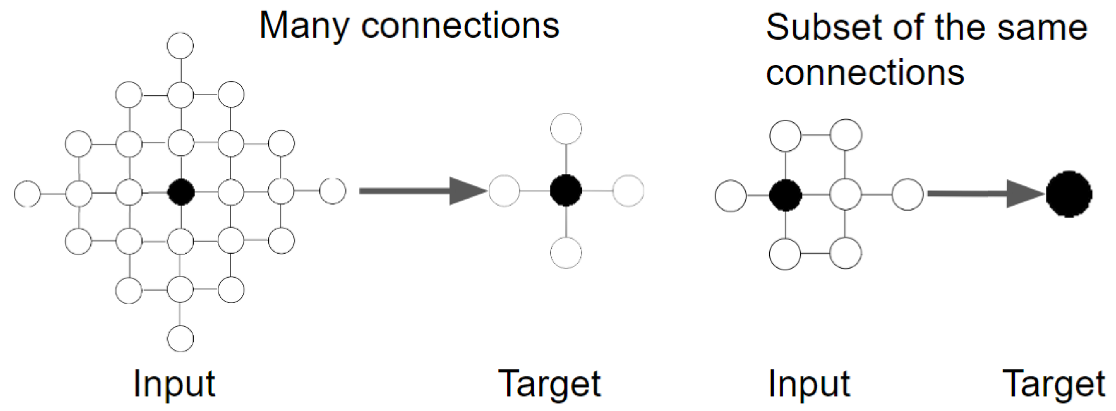
- But larger input can be computationally expensive

Proposed Approach

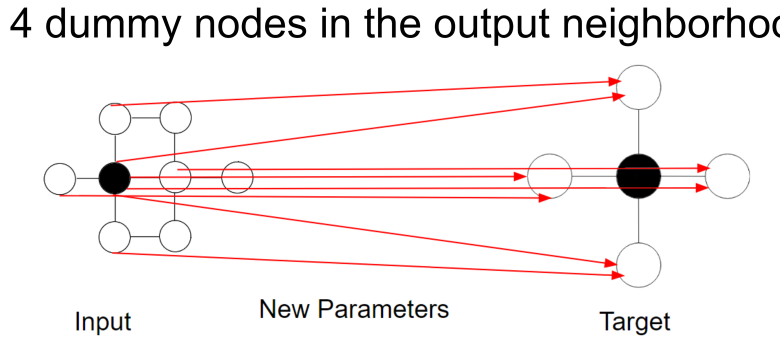
- Approximate inference on large SD-BNA, using a smaller **surrogate** SD-BNA
- Extend smaller SD-BNA with new parameters learned using samples from the large SD-BNA

Surrogate Maximum Likelihood Models

- Consider a 25 node input SD-BNA (B) and an 8 node input SD-BNA (A)
- Smaller network is a subset of the larger network



- Add 4 dummy nodes in the output neighborhood



- New edges from input the target
- Each new node is conditioned on two nodes in the input, CPTs of the form $\theta_{x,y,z} = p_{x|yz}$
- To learn these CPTs, we use MLE with samples from larger SD-BNA

Basic Maximum Likelihood Estimation (MLE) algorithm

1. Forward sample larger model for full configuration of every variable in the network
2. The 8 node network is fully observed for each data point since its variables are subsumed in the large network
3. We add symmetry constraints using pre-computed quantities from non-extended small network.
4. Final problem is to maximize the following log-likelihood expression:

$$\theta^* = \arg\max_{\theta} \sum_{j=1}^{L} \log \theta(Nb(X^{(j)})_{t+1}, X_t^{(j)}, Nb(X^{(j)})_t)$$

subject to

$$p_{s_i}(t+1) = \sum_{s_j, s_k \in S} \theta_{s_i,s_j,s_k} \cdot p_{s_j s_k}(t), \quad \forall s_i \in S$$

KL-Search Minimization

- Instead of random samples, combine search with sampling to get samples that reduce KL-divergence
- Search OR-search tree of large network using best-first heuristic:

$$h_{kl}(X_1{=}0) = [\log(P_A(X_1{=}0)) - \log(P_B(X_1{=}0))] \cdot P_B(X_1{=}0)$$

- $P_A(X_1 = 0)$ difficult to compute, approximate with Weight Mini-bucket Elimination with small i-bound (KL-Search) or single sample Monte Carlo estimate (Fast KL-Search)

KL-Search

Algorithm 1: KL-Search Minimization

Input: Two Bayesian networks: a large network A, a smaller network B, and a parameterized extended network B_θ such that all nodes in B are in A ($B \subset A$), a variable ordering o over A, initial distribution $p_{yz}(t)$, and pre-computed output distribution $p_x(t+1)$

Parameters: Number of samples L

Output: θ, estimated parameters that minimize difference between A and B_θ

```
T ← the OR-search tree on A using ordering o;
OPEN ← {⟨root(T), 0⟩};
// frontier nodes are ordered by the 2nd value
for i = 1 → L do
    v ← OPEN.dequeue() ;    // remove the node
      of highest priority
    for u ∈ children(v) do
        h_kl(u) ← [log(P_A(u)) − log(P_B(u))] · P_B(u);
        Append < u, h_kl(u) > to OPEN;
    end
end
Let X be an empty list;
for v ∈ OPEN do                    // leaf nodes
    Forward sample x, a full configuration of A
      conditioned on the partial configuration
      represented by v;
    Append x to X;
end
Solve θ* = arg max_θ Σ_{j=1}^{L} log P_{B_θ}(x^j) · P_A(x^j),
  subject to
p_{s_i}(t+1) = Σ_{s_j,s_k∈S} θ_{s_i,s_j,s_k} · p_{s_j s_k}(t),  ∀s_i ∈ S;
Return θ*;
```

Convergence of KL-Search

Theorem 6.0.1 (Asymptotic Convergence of KL-Search Minimization). *Let θ_L be the result of KL-Search Minimization [Algorithm 1] given L samples. Then given a family of extended networks B_θ parameterized by θ:*

$$\lim_{L \to \infty} \theta_L = \arg\min_{\theta} D_{KL}(P_A || P_{B_\theta})$$

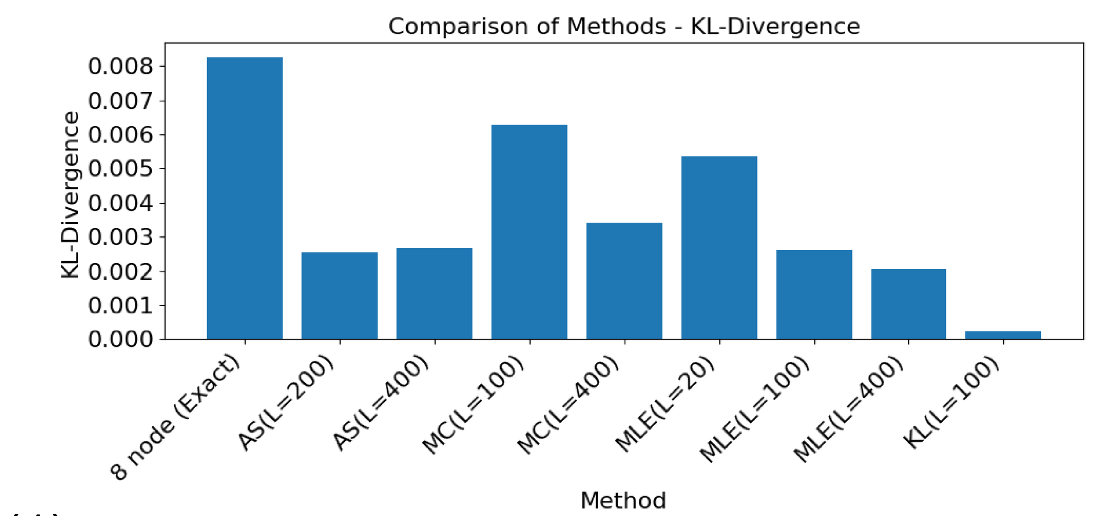
Experimental Setup

- Compare with existing approaches such as Pair Approximation and Abstraction Sampling (AS)
- Task: estimate joint probability distribution of nodes in output distribution (e.g. nodes in red area at each iteration)
- Use KL-divergence between estimate generated by simulation (ground truth) and estimate from methods begin compared.

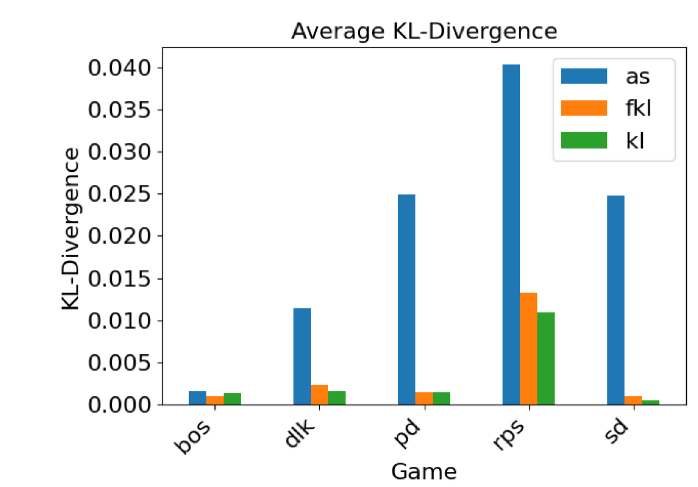
$$D_{KL}(P_{sim}(X_1|Nb(X)_1)||P_{method}(X_1|Nb(X)_1))$$

Results

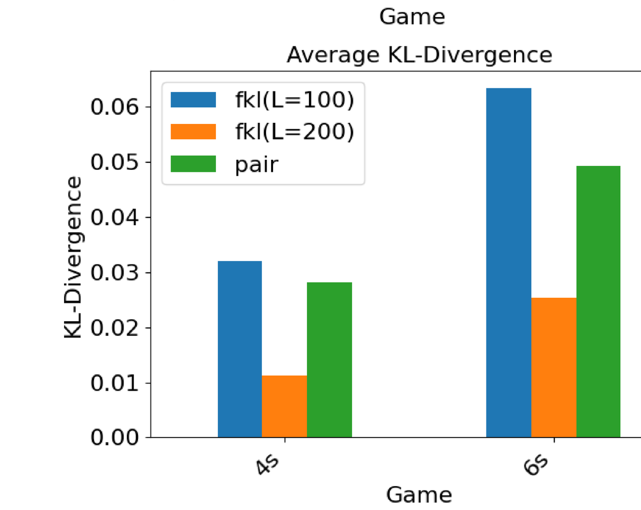
(1)



(2)



(3)



(1)KL-Search outperforms other methods by a considerable amount on the Deadlock game

(2)KL-Search and Fast KL-Search outperform Abstraction Sampling in several 2 and 3 strategy evolutionary games

(3)Fast KL-Search outperforms pair approximation on games with more than 4 strategieswhen given more than 100 samples

Time per sample/probe

$\|S\|$	AS	KL-Search	Fast KL-Search
2	0.1175	0.01419	0.01624
3	0.2203	0.03063	0.03725
4	-	-	0.1630
6	-	-	0.7299

Fast KL-Search can be applied to high strategy games since it does not need to compute WMBE.

Future Research

- Apply to domains with high degree of symmetry beyond spatial evolutionary games

Acknowledgements

This work supported in part by NSF grant IIS-2008516 and AFOSR grant 1010GWA357.