Approximating Spatial Evolutionary Games using Bayesian Networks

Vincent Hsiao[1], Dana Nau[1], Xinyue Pan[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 = (A, S, U, G, F, γ, μ)
  - A - set of *M* agents, S - set of strategies
  - U - payoff matrix
  - G - graph of population structure
    - N(i) - neighborhood of agent i
  - F - replicator rule (e.g. Fermi rule)

U =

	s_1	s_2
s_1	a	b
s_2	c	d

Interaction Phase

- Each agent A_i can play some strategy $s_i \in S$ and receive payoff π_i

$$\pi_i = \sum_{j \in N(i)} U[s_i, s_j]$$

Update Phase

- Percentage of agents γ use rule F to update their strategies based on the payoffs received and neighbor's payoffs

$$\Pr_f(\pi, \pi') = \frac{1}{(1 + e^{-s(\pi' - \pi)})}$$

- Small probability μ of mutating to a random strategy

T iterations: interaction phase, update phase

Problem Statement

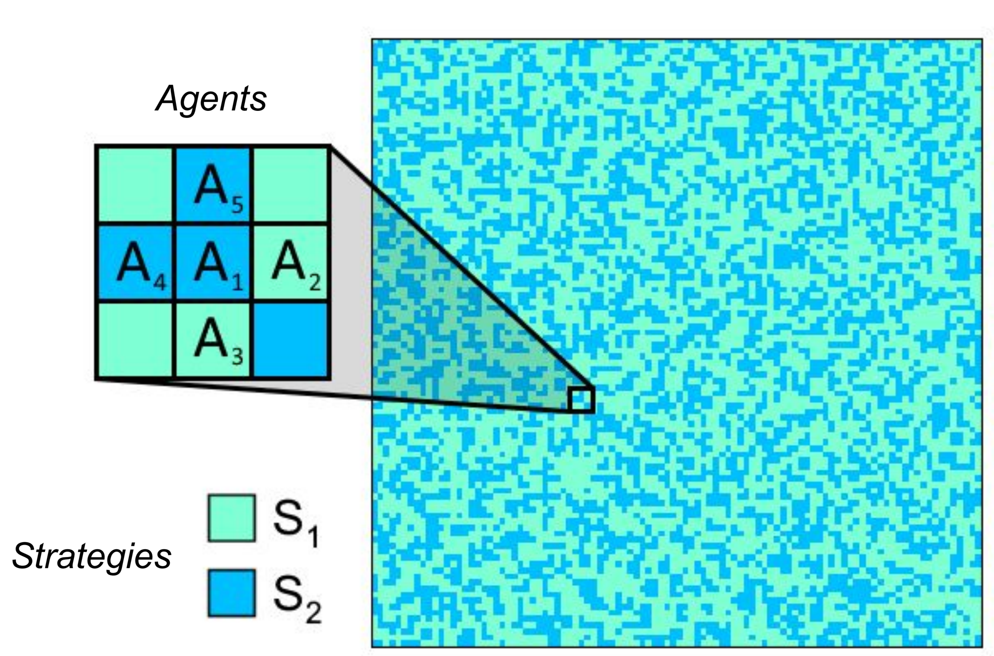
Current Approach

- Evaluate using agent-based Monte-Carlo simulations
  - Difficult to validate
  - Need to be repeated many times
- Alternative methods such as pair approximation
  - Not very accurate

Proposed Approach

- Model using Dynamic Bayesian Networks (DBN)
- Approximate the spatial evolutionary game through the DBN truncation by exploiting symmetry
  - Better accuracy than pair approximation with respect to stochastic simulations.

Spatial EGT Model



Dynamic Bayesian Network Model

Exact Model

We define a Dynamic Bayesian Network (DBN) that fully captures our spatial evolutionary game.

Given a spatial EGT = (A, S, U, G, F, γ, μ), the DBN (X(t), D(t), P(t)) is defined as follows:

The variable set X(t) = A(t) ∪ Pay(t):

- $A_i(t)$: $S_i(t)$, the strategy of agent A_i at each iteration t
- $Pay_i(t)$: the payoff received by the agent A_i during the interaction phase at time t.

The probability functions P(t) are defined:

For a payoff variable

$$\Pr(Pay_i(t) \mid A_i(t), N(A_i(t))) = \begin{cases} 1 & \text{if } Pay_i(t) = \sum_{j \in N(i)} U(A_i(t), A_j(t)) \\ 0 & \text{otherwise} \end{cases}$$

For a strategy variable

- $\Pr(A_i(t+1) \mid parents)$ can be expressed as a decision tree. For example, with the Fermi rule:
  - update: did an update happen?
  - mut: did mutation happen?
  - rand: which neighbor was chosen?
- Example: if (update = 1) and (mut = 0):

$$\Pr(A(t+1)_i = s_{t+1} \mid A_i(t) = s_t, \text{other parents}) = \sum_{j \in N(i)} \frac{1}{d} \Pr_f(Pay_i, Pay_j) \Pr_\delta (1 - \Pr_\varnothing) + \Pr_\varnothing$$

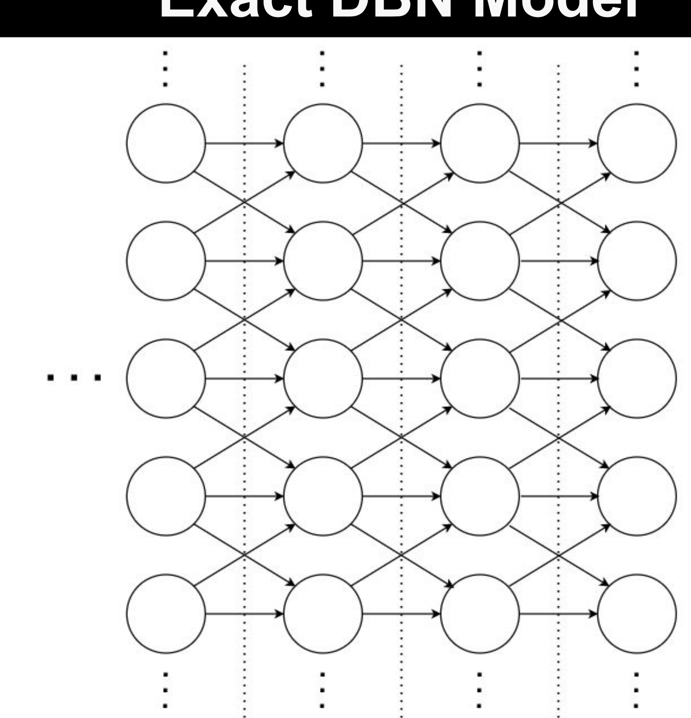
where

$$\Pr_\delta = \mathbb{1}_{A_i(t+1) = A_j(t)}, \Pr_\varnothing = \mathbb{1}_{A_i(t+1) = A_i(t)}$$

Evaluation

- Can use DBN tools to evaluate
  - Message passing inference
- Exact inference can be computationally expensive
  - Solution: we can exploit symmetry
- Proposal: approximate by truncation
  - Convert from DBN to iterative 2-timestep BNs

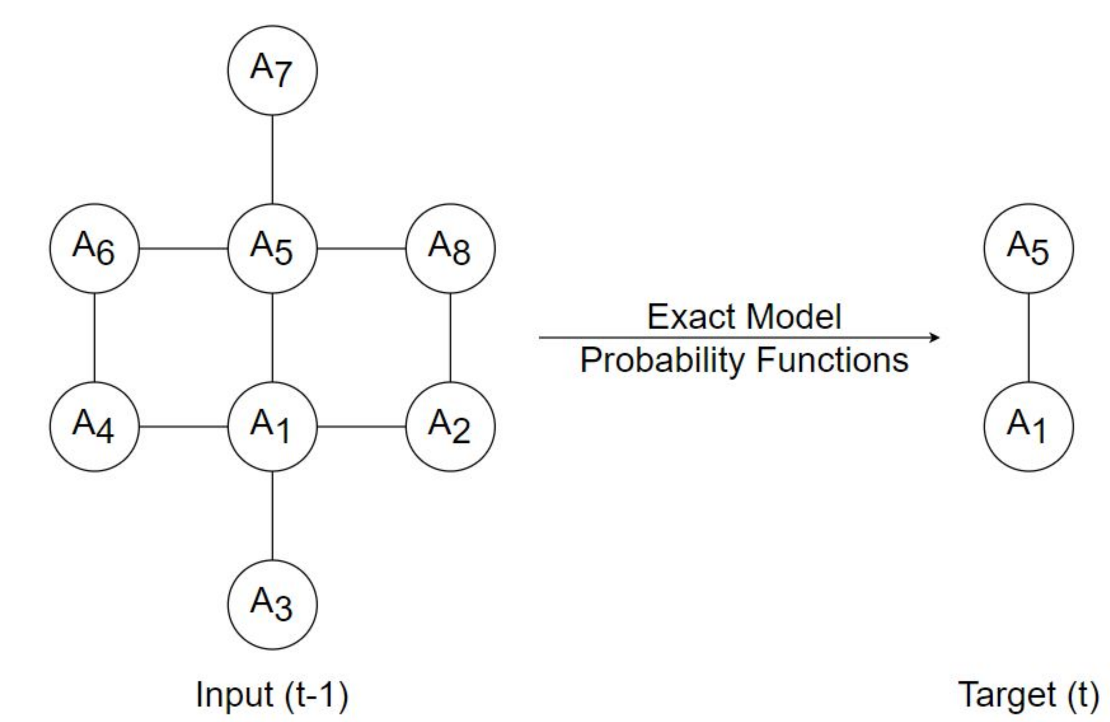
Exact DBN Model



Truncation Approximation

Truncation Neighborhood

- Choose subset of agent nodes as input neighborhood
- Construct a 2-timestep Bayesian Network (BN) that takes nodes in input neighborhood to target neighborhood using CPTs from exact model
- Target neighborhood may consist of only one or two nodes

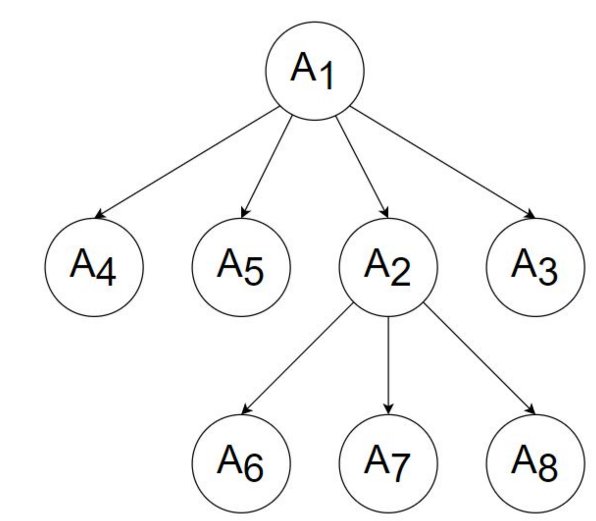


Output query

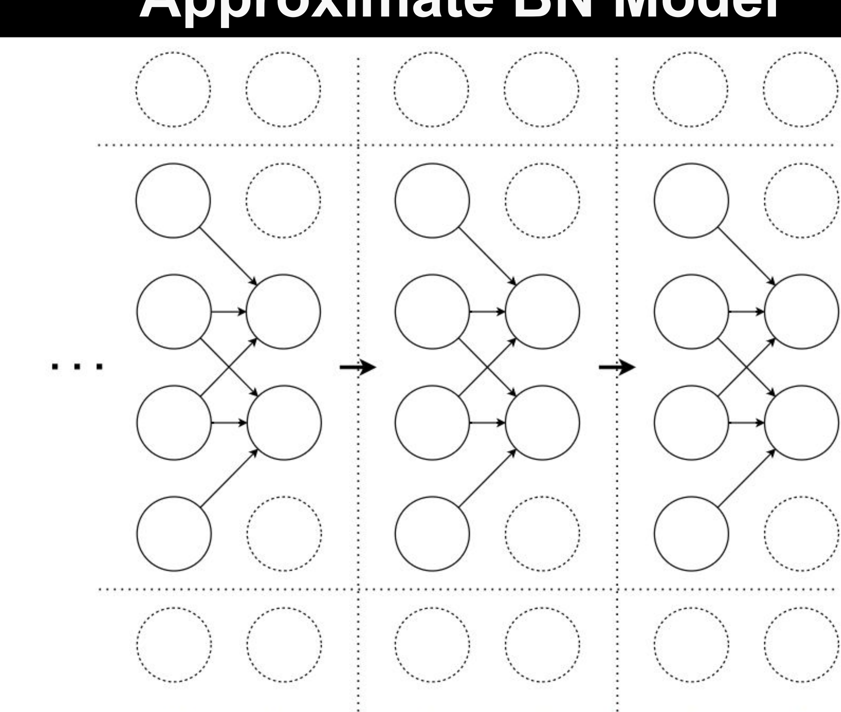
- Query a selection of lower order distributions from target neighborhood

Input definition

- 2-timestep BNs are not connected like DBN
  - Joint distribution of input neighborhood at next timestep is unknown
- We use a probability tree approximating the input neighborhood using distributions from previous output



Approximate BN Model

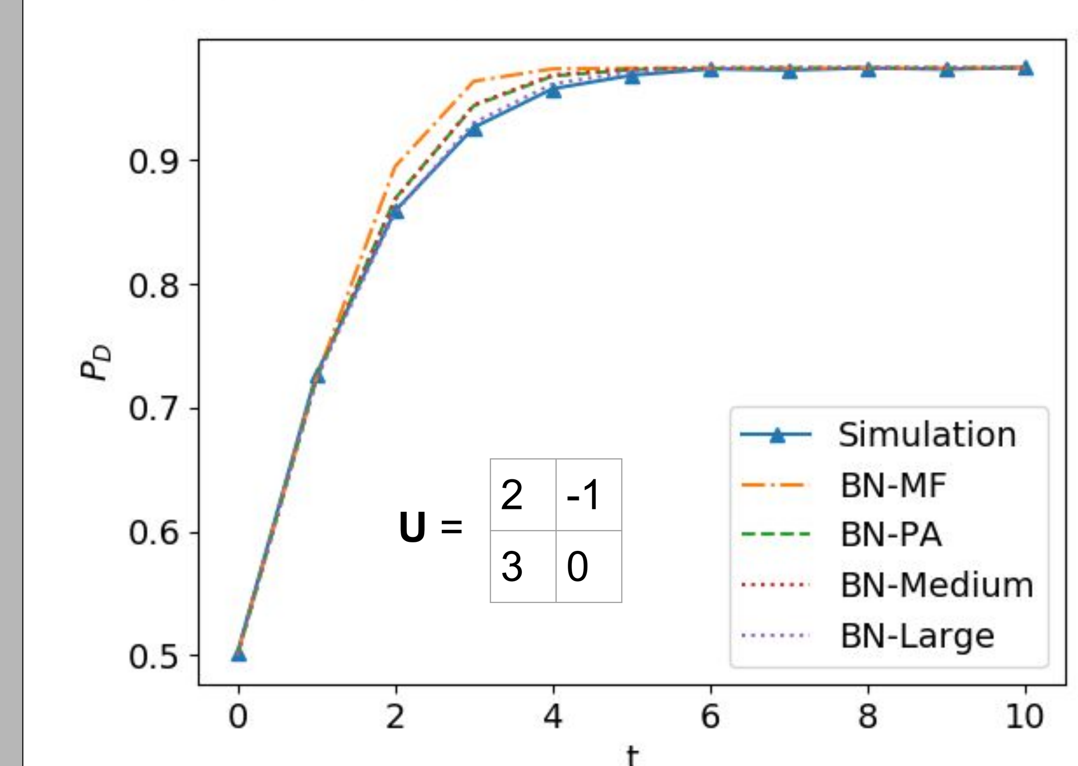


Results

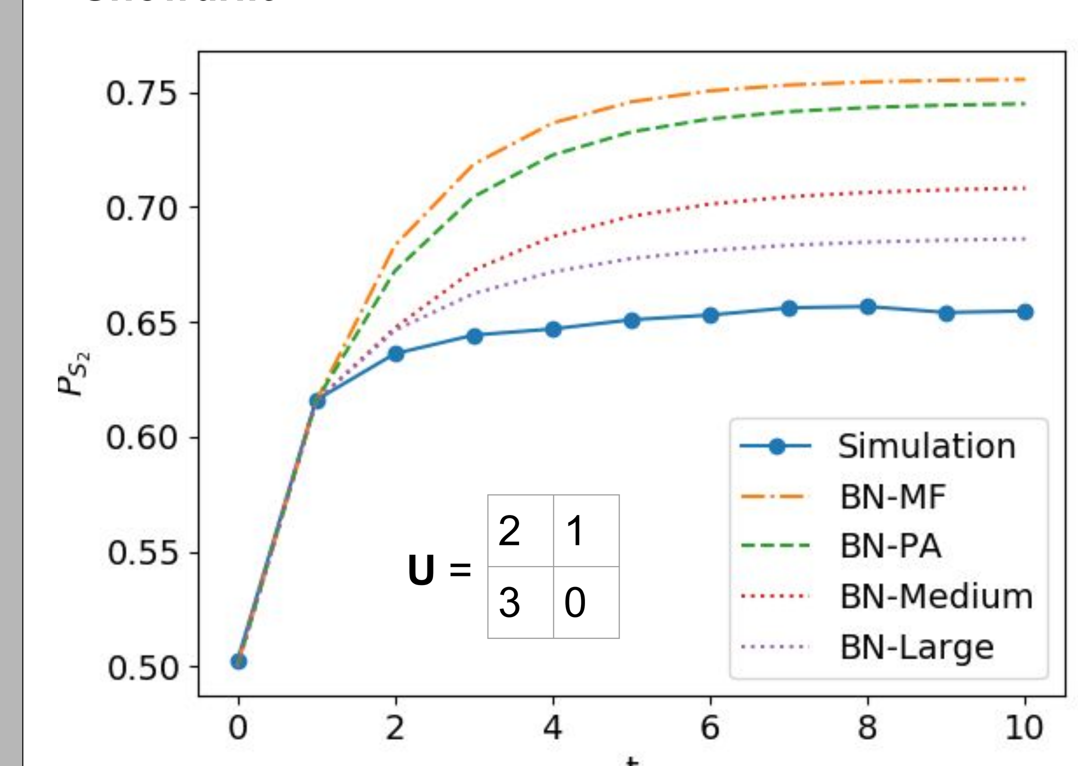
Experimental Setup

- Compare with average of 20 agent-based simulations on a 50 x 50 grid
- Four different levels of approximation:
  - BN-MF: 8 nodes (without tree approximation)
  - BN-PA: 8 nodes
  - BN-Medium: 13 nodes
  - BN-Large: 25 nodes

Prisoner's Dilemma



Snowdrift



- Larger approximation neighborhoods reduce error
- Error is reduced even in cases such as snowdrift where pair approximation does not have good quantitative agreement with simulation results

Future Research

- Tune approximation parameters to balance accuracy and complexity
- Explore impact of approximate inference algorithms

Acknowledgements

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