An Algorithm for Probabilistic Alternating Simulation

Authors:

Chenyi Zhang University of New South Wales

Jun Pang University of Luxembourg

Presentation:

Tim Muller University of Luxembourg

Goal

- Introduce games and simulations thereon.
  - Relevant simulation is PA-I-simulation.
- Introduce General Coarsest Partition Problem.
- Link between GCPP and PA-I-simulation.
- Solve the GCPP with polynomial algorithm.
  - Obtain largest probabilistic alternating simulation relation.

Example of a game

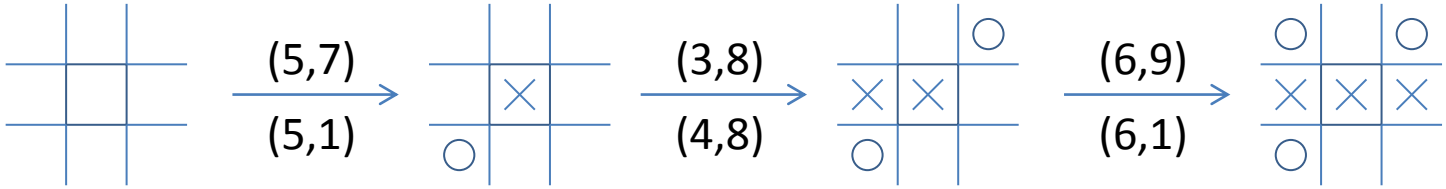
- Probabilistic simultaneous tic-tac-toe:
  - Similar to tic-tac-toe:
    - Board consists of nine squares.
    - At most one player can mark a square.
    - Three in a row wins (vertical, horizontal or diagonal).
  - Turns are simultaneous.
  - A player picks a favorite and an alternative square:
    - If the favorites differ, both players mark their favorite.
    - If the favorites are the same, each player has a 50% chance of marking their favorite, the other player marks his alternative.
  - If there is only one square left, and there is no winner, then the game is over, and it's a draw.

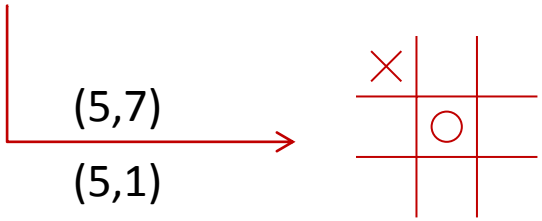
Example run

○ Player I

× Player II

1	2	3
4	5	6
7	8	9





Probabilistic game structure

- A probabilistic game structure consists of:
  - A finite set of states S, with an initial state s_0.
  - A finite set of joint actions $Act = Act_{\mathrm{I}} \quad Act_{\mathrm{II}}$,
    - Act_{I} is the set of actions for player I.
  - A labeling function $L : S \rightarrow 2^{Prop}$
  - A transition function $\delta : S \quad Act \rightarrow \mathcal{D}(S)$

Example (ctd.)

- A state is a board with equal number of O's and X's, and squares marked at most once.
  - Initial state is the empty board.
- An action for one player is any pair (n,m), with $1 \leq n, m \leq 9$ and $n \neq m$.
- Predicate P_1 holds iff there is a line of three O's, and P_2 holds iff there is a line of three X's.
- The pair s, $((n,m), (n',m'))$ maps to U(s) if n, m, n' or m' is marked in s or if P_1 or P_2 holds. Otherwise:
  - The pair s, $((n,m), (n',m'))$ maps to U(s[O on n; X on n']) if $n \neq n'$.
  - The pair s, $((n,m), (n,m'))$ maps to U(s[O on n; X on m'], s[O on m; X on n]).

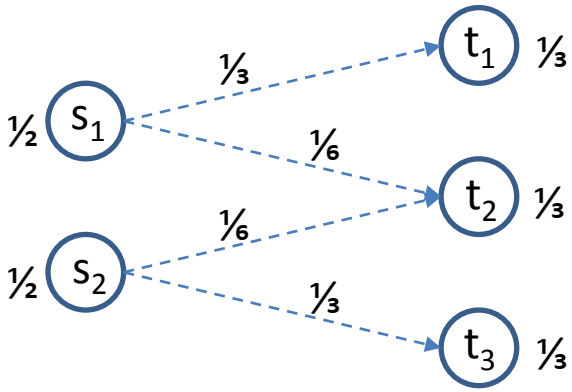
Mixed actions

- A mixed action of player I is a distribution over Act_{I}, denoted with $\pi \in \Pi_{\mathrm{I}}$.

$$\bar{\delta}(s,(\pi_1, \pi_2))(t) = \sum a_1 \in Act_{\mathrm{I}}, a_2 \in Act_{\mathrm{II}} : \pi_1(a_1) \cdot \pi_2(a_2) \cdot \delta(s,(a_1,a_2))(t)$$

Simulation

- If A simulates B, then every move by B can be matched by A.
  - Probabilistic:
    - From R to $\overline{R}$

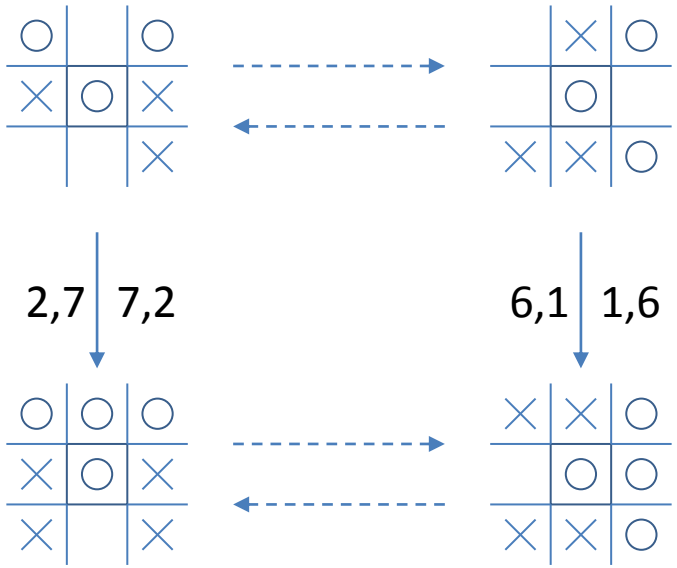


  - Alternating:
    - If A I-simulates B, then every move of I in B can be matched by a move of I in A, such that for every counter-move of II in A there is a counter-move of II in B.

Probabilistic alternating I-simulation

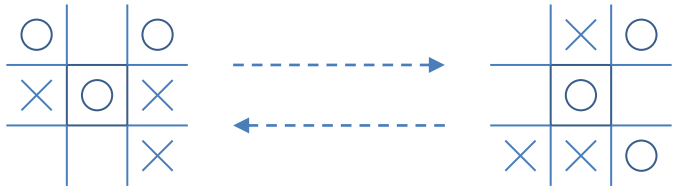
- A PA-I-simulation is a relation $\sqsubseteq \subseteq S \quad S$ satisfying: if $s \sqsubseteq t$, then:
  - $L(s) = L(t)$
  - For all $\pi_1 \in \Pi_{\mathrm{I}}$, there exists $\pi_1' \in \Pi_{\mathrm{I}}$, such that for all $\pi_2' \in \Pi_{\mathrm{II}}$, there exists $\pi_2 \in \Pi_{\mathrm{II}}$, such that $\delta(s,(\pi_1,\pi_2)) \bar{\sqsubseteq} \delta(t,(\pi_2', \pi_2'))$
- If $s \sqsubseteq t$ and $t \sqsubseteq s$, then s and t are PA-I-simulation equivalent.

Examples of PA-I-simulations

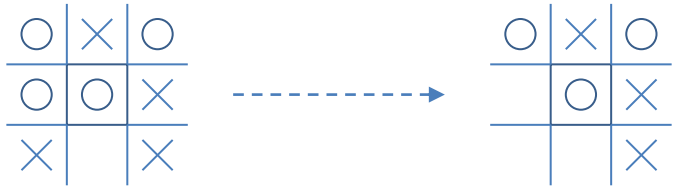


PA-I-simulation equivalent!

Examples of PA-I-simulations



PA-I-simulation equivalent!



4,8 ?,?

Tie

All ties, wins and losses are PA-I-simulation equivalent.

General Coarsest Partition Problem

- We want to partition a set of states into blocks, where all states in a block are PA-I-simulation equivalent.
- We want to define a partial order, $\preceq$, such that if $P \preceq Q$, then every state in Q simulates every state in P.
- We want to find the coarsest such partition and partial order (partition pair).

From GCPP to PA-I-simulation

- We write $[s]_\Sigma$ for the block containing s.
- Define a relation $\trianglelefteq_{(\Sigma,\preceq)} \subseteq S \quad S$ as $s \trianglelefteq_{(\Sigma,\preceq)} t$ iff $[s]_\Sigma \preceq [t]_\Sigma$
- If $(\Sigma, \preceq)$ is the coarsest partition pair, then $s \trianglelefteq_{(\Sigma,\preceq)} t$ is the largest PA-I-simulation.
  - There is a unique coarsest partition pair.

Sketch of approach

- Start with a partition pair that is too coarse.
  - If $s \sqsubseteq t$ then $[s]_\Sigma \preceq [t]_\Sigma$
- Keep refining blocks in the partition pair, until nothing changes.
  - Make sure not to refine too much!
- Return the partition pair.

Main procedure

Algorithm 2 Computing the Generalised Coarsest Partition Pair

INPUT: a probabilistic game structure $\mathcal{G} = \langle S, s_0, \mathcal{L}, Act, \delta \rangle$
OUTPUT: a partition pair $(\Sigma, \preceq)$ on S
function GCPP $(\mathcal{G})$
  $\Sigma := \{\{t \mid \mathcal{L}(t) = \mathcal{L}(s)\} \mid s \in S\}$; $\preceq := \{(B, B) \mid B \in \Sigma\}$
  $\Sigma' := \emptyset$; $\preceq' := \emptyset$
  while $\Sigma \neq \Sigma' \vee \preceq \neq \preceq'$ **do**
    $\Sigma' := \Sigma$; $\preceq' := \preceq$
    for each $B \in \Sigma$ **do**
      $(\Sigma_B, \preceq_B) := \mathsf{Split}((\Sigma', \preceq'), B)$
      $\Sigma := \Sigma \setminus \{B\} \cup \Sigma_B$
      $\preceq := \preceq \cup \preceq_B$
        $\cup \{(B', X) \mid X \in \Sigma : B' \in \Sigma_B : (B, X) \in \preceq\}$
        $\cup \{(X, B') \mid X \in \Sigma : B' \in \Sigma_B : (X, B) \in \preceq\}$
        $\setminus \{(B, X), (X, B) \mid X \in \Sigma : (X, B), (B, X) \in \preceq\}$
    endfor
  endwhile
  return $(\Sigma, \preceq)$

Sketch of refining blocks

- Input is a partition pair $(\Sigma, \preceq)$ and a block B E.
- Start with a partition pair $(\Sigma_B, \preceq_B)$ that is to fine.
- Keep merging blocks until $(\Sigma_B, \preceq_B)$ stops changing:
  - For any pair of blocks:
    - If a state s in A can simulate a state t in B under $(\Sigma, \preceq)$, and vice versa, then A and B should be the same block.
    - If a state s in A can simulate a state t in B under $(\Sigma, \preceq)$, but not vice versa, then $A \preceq B$ should hold (and vice versa).
- Return $(\Sigma_B, \preceq_B)$

Refinement procedure

Algorithm 1 Refining a block to make it stable on a partition pair

INPUT: a partition pair $(\Sigma, \preceq)$, a block $B \in \Sigma$
OUTPUT: a partition pair $(\Sigma_B, \preceq_B)$ on B
function Split $((\Sigma, \preceq), B)$
  $\Sigma_B := \{\{s\} \mid s \in B\}$; $\preceq_B := \{(s,s) \mid s \in B\}$; $\Sigma' := \emptyset$; $\preceq' := \emptyset$
  while $\Sigma_B \neq \Sigma' \vee \preceq_B \neq \preceq'$ **do**
    $\Sigma' := \Sigma_B$; $\preceq' := \preceq_B$
    for each *distinct* $B_1, B_2 \in \Sigma_B$ **do**
      pick any $s_1 \in B_1$ *and* $s_2 \in B_2$
      if (CanSim$((\Sigma, \preceq), s_1, s_2) \wedge$ CanSim$((\Sigma, \preceq), s_2, s_1)$) **then**
        $\Sigma_B := \Sigma_B \setminus \{B_1, B_2\} \cup \{B_1 \cup B_2\}$
        $\preceq_B := \preceq_B \cup \{(X, B_1 \cup B_2) \mid X \in \Sigma : (X, B_1) \in \preceq_B \vee (X, B_2) \in \preceq_B\}$
            $\cup \{(B_1 \cup B_2, X) \mid X \in \Sigma : (B_1, X) \in \preceq_B \vee (B_2, X) \in \preceq_B\}$
            $\setminus \{(B_i, X), (X, B_i) \mid X \in \Sigma : (B_i, X), (X, B_i) \in \preceq_B \wedge i \in \{1,2\}\}$
      else if (CanSim$((\Sigma, \preceq), s_1, s_2)$) **then**
        $\preceq_B := \preceq_B \cup \{(B_2, B_1)\}$
      else if (CanSim$((\Sigma, \preceq), s_2, s_1)$) **then**
        $\preceq_B := \preceq_B \cup \{(B_1, B_2)\}$
    endfor
  endwhile
  return $(\Sigma_B, \preceq_B)$

Correctness

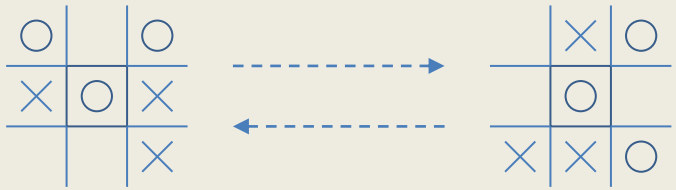
- Define an operator ρ on partition pairs, by letting $\rho((\Sigma, \preceq))$ be the coarsest partition pair $(\Sigma', \preceq')$, finer than $(\Sigma, \preceq)$ that is stable on $(\Sigma, \preceq)$.
  - If $(\Sigma_1, \preceq_1)$ is finer than $(\Sigma_2, \preceq_2)$, then $(\Sigma_1, \preceq_1)$ is stable on $(\Sigma_1, \preceq_1)$, when for all $P, Q \in \Sigma_1$ with $P \preceq_1 Q$ and $s \in P$ and $t \in Q$, $s \sqsubseteq_{(\Sigma_2, \preceq_2)} t$.
- Running Split on each block in Σ, for $(\Sigma, \preceq)$ is equal to $\rho((\Sigma, \preceq))$.
- Theorem 1: If $(\Sigma, \preceq)$ is the fixed point of $\rho((\Sigma_0, Id))$ then $\sqsubseteq_{(\Sigma, \preceq)}$ is the largest PA-I-simulation.

Polynomial run-time

- CanSim can be reduced to $|Act_{\mathrm{I}}|$ applications of CanFollow.
- CanFollow can be reduced to a linear program (Lemma 9).
- The foreach in split is trivially polynomial.
- For the while in split, note that there are at most O(number of blocks) iterations.
- The same argument applies to the main GCPP algorithm.

Example (ctd.)

- State space reduction:



Block

- Winning strategy:

Conclusion

- Probabilistic game structures are a relevant (interesting) class of structures.
- PA-I-simulations offer reductions that help answer (some) questions about a PGS.
- There is a polynomial algorithm that finds the strongest PA-I-simulation.

Questions?