Algorithms for Model Checking (2IW55)

Lecture 12

The Recursive Algorithm for Parity games

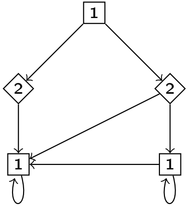
Background material:

"Recursive Solving of Parity Games Requires Exponential Time", Oliver Friedmann

December 31, 2010

Parity games (recap)

Identify graph, priorities, owners, plays, and strategies in the following parity game.



Outline

Minimizing parity games

Solving parity games

Transformations on parity games

- Self-loop elimination (vs Local resolution)
- Priority compaction
- Priority propagation
- Bisimulation minimisation

Bisimulation

Definition (Bisimilarity of vertices)

Let $G = (V, E, \mathrm{p}, (V_\diamond, V_\Box))$ be a parity game. Let R be a symmetric relation. R is a bisimulation relation if $v \, R \, v'$ implies

- $v \in V_\diamond \Leftrightarrow v' \in V_\diamond$
- $\mathrm{p}(v) = \mathrm{p}(v')$
- $v \to w$ implies $\exists w'$ such that $v' \to w'$ and $w \, R \, w'$

Vertices v and v' are bisimilar ($v \equiv v'$) iff there exists a bisimulation relation R such that $v \, R \, v'$.

Bisimulation

Definition (Bisimilarity of vertices)

Let $G = (V, E, \mathsf{p}, (V_\Diamond, V_\Box))$ be a parity game. Let R be a symmetric relation. R is a bisimulation relation if $v \, R \, v'$ implies

- $v \in V_\Diamond \Leftrightarrow v' \in V_\Diamond$
- $\mathsf{p}(v) = \mathsf{p}(v')$
- $v \to w$ implies $\exists w'$ such that $v' \to w'$ and $w \, R \, w'$

Vertices v and v' are bisimilar ($v \equiv v'$) iff there exists a bisimulation relation R such that $v \, R \, v'$.

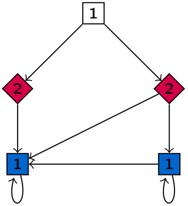
Theorem

$v \equiv v'$ implies that v and v' are won by the same player

TU/e Technische Universiteit Eindhoven University of Technology

Bisimilarity (example)

Original



Minimal bisimilar parity game

Outline

Minimizing parity games

Solving parity games

Goal

Let $G = (V, E, p, (V_\diamond, V_\Box))$ be a parity game.

- There is a unique partition $(W_\diamond, W_\Box)$ of V such that:
  - $\diamond$ has winning strategy $\varrho_\diamond$ from $W_\diamond$, and
  - $\Box$ has winning strategy $\varrho_\Box$ from $W_\Box$.

Goal of parity game algorithms

Compute partitioning $(W_\diamond, W_\Box)$ with strategies $\varrho_\diamond$ and $\varrho_\Box$ of V, such that $\varrho_\diamond$ is winning for player $\diamond$ from $W_\diamond$ and $\varrho_\Box$ is winning for player $\Box$ from $W_\Box$.

Notation

Let $G = (V, E, p, (V_\diamond, V_\Box))$ be a parity game.
We use the following notation:

- $\overline{\diamond}$ is $\Box$, $\overline{\Box}$ is $\diamond$
- $G \setminus U$ is parity game G restricted to the vertices outside U. Formally $G \setminus U = (V', E', p', (V'_\diamond, V'_\Box))$, with
  - $V' = V \setminus U$,
  - $E' = E \cap (V \setminus U)^2$,
  - $p'(v) = p(v)$ for $v \in V \setminus U$,
  - $V'_\diamond = V_\diamond \setminus U$, and
  - $V'_\Box = V_\Box \setminus U$

Recursive algorithm (intuition)

- Divide and conquer
- Base: empty game
- Step: assemble winning sets/strategies from
  - winning sets/strategies of subgames
  - attractor strategy for one of players reaching set of nodes with minimal priority in the game

Attractor sets

The attractor set for $\bigcirc$ and set $U \subseteq V$ is the set of vertices such that $\bigcirc$ can force any play to reach U.

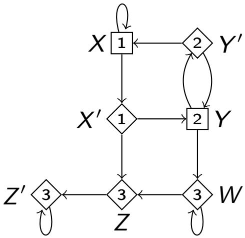
Definition

Let $U \subseteq V$. We define the attractor sets inductively as follows:

$$\begin{aligned}
Attr^{0}_{\bigcirc}(G, U) &= U \\
Attr^{k+1}_{\bigcirc}(G, U) &= Attr^{k}_{\bigcirc}(G, U) \\
&\cup \{v \in V_{\bigcirc} \mid \exists v' \in V : (v, v') \in E \wedge v' \in Attr^{k}_{\bigcirc}(G, U)\} \\
&\cup \{v \in V_{\overline{\bigcirc}} \mid \forall v' \in V : (v, v') \in E \implies v' \in Attr^{k}_{\bigcirc}(G, U)\} \\
Attr_{\bigcirc}(G, U) &= \bigcup_{k \in \mathbb{N}} Attr^{k}_{\bigcirc}(G, U)
\end{aligned}$$

Example

Consider parity game G:



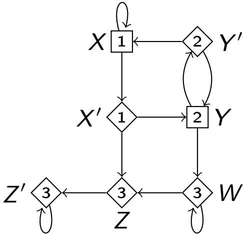
Compute:

- $Attr_{\diamond}(G, \{Z\})$
- $Attr_{\square}(G, \{W\})$

Example of attractor sets

Example

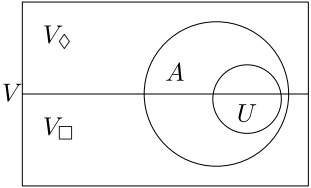
Consider parity game G:



Compute:

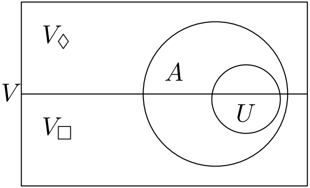
- $Attr_{\diamond}(G, \{Z\}) = \{Z, X', W\}$
- $Attr_{\Box}(G, \{W\}) = \{W, Y\}$

Observations



Let $U \subseteq V$. Let $A = Attr_\Diamond(G, U)$.

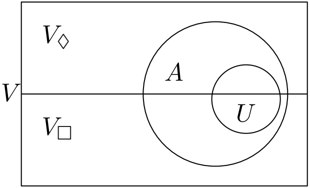
Observations



Let $U \subseteq V$. Let $A = Attr_\Diamond(G, U)$.

- $\Diamond$ cannot escape from $V \setminus A$.

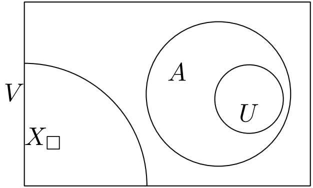
Observations



Let $U \subseteq V$. Let $A = Attr_\Diamond(G, U)$.

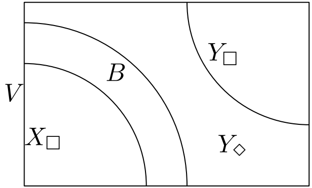
- $\Diamond$ cannot escape from $V \setminus A$.
- $\Box$ cannot escape from A.

Observations

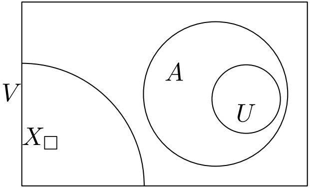


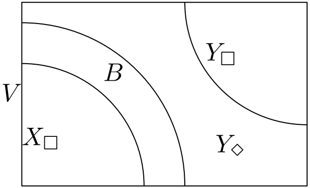
Let $U \subseteq V$. Let $A = Attr_\diamond(G, U)$.
Assume:

- $X_\square$ is winning set for $\square$ on $G \setminus A$;
- $B = Attr_\square(G, X_\square)$;
- $Y_\diamond$ is winning set for $\diamond$ on $G \setminus B$;
- $Y_\square$ is winning set for $\square$ on $G \setminus B$.



Observations





Let $U \subseteq V$. Let $A = Attr_\Diamond(G, U)$.
Assume:

- $X_\Box$ is winning set for $\Box$ on $G \setminus A$;
- $B = Attr_\Box(G, X_\Box)$;
- $Y_\Diamond$ is winning set for $\Diamond$ on $G \setminus B$;
- $Y_\Box$ is winning set for $\Box$ on $G \setminus B$.

Then:

- Player $\Diamond$ can never leave B;
- Player $\Box$ can never leave $V \setminus B$;
- A winning strategy for player $\Box$ in $G \setminus (V \setminus B)$ from $V_\Box \cap B$ is also a winning strategy for player $\Box$ in G from $V_\Box \cap B$.

Recursive algorithm (McNaughton '93, Zielonka '98)

Recursively solve a parity game: *Recursive*(G). Returns partitioning $(W_\Diamond, W_\Box)$ such that $\Diamond$ wins from $W_\Diamond$, and $\Box$ wins from $W_\Box$.

if $V_G = \emptyset$ **then**
    $W_\Diamond \leftarrow \emptyset$
    $W_\Box \leftarrow \emptyset$
    return $(W_\Diamond, W_\Box)$
end if

Technische Universiteit Eindhoven University of Technology

Recursive algorithm (McNaughton '93, Zielonka '98)

Recursively solve a parity game: $\mathit{Recursive}(G)$. Returns partitioning $(W_\diamond, W_\Box)$ such that $\diamond$ wins from $W_\diamond$, and $\Box$ wins from $W_\Box$.

```
if V_G = ∅ then
    W_◇ ← ∅
    W_□ ← ∅
    return (W_◇, W_□)
end if
m ← min{p(v) | v ∈ V}
(* Paper: max *)
◯ ← { ◇  if m is even
    { □  otherwise
U ← {v ∈ V | p(v) = m}
A ← Attr_◯(G, U)
(X_◇, X_□) ← Recursive(G \ A)
if X_◯̄ = ∅ then
    W_◯ ← A ∪ X_◯
    W_◯̄ ← ∅
```

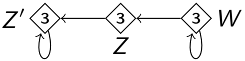

Technische Universiteit Eindhoven University of Technology

Recursive algorithm (McNaughton '93, Zielonka '98)

Recursively solve a parity game: *Recursive*(G). Returns partitioning $(W_\Diamond, W_\Box)$ such that $\Diamond$ wins from $W_\Diamond$, and $\Box$ wins from $W_\Box$.

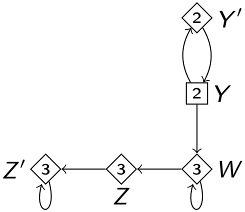
if $V_G = \emptyset$ **then**
  $W_\Diamond \leftarrow \emptyset$
  $W_\Box \leftarrow \emptyset$
  return $(W_\Diamond, W_\Box)$
end if
$m \leftarrow \min\{p(v) \mid v \in V\}$
(* Paper: max *)
$\bigcirc \leftarrow \begin{cases} \Diamond & \text{if m is even} \\ \Box & \text{otherwise} \end{cases}$
$U \leftarrow \{v \in V \mid p(v) = m\}$
$A \leftarrow Attr_\bigcirc(G, U)$
$(X_\Diamond, X_\Box) \leftarrow Recursive(G \setminus A)$
if $X_{\overline{\bigcirc}} = \emptyset$ **then**
  $W_\bigcirc \leftarrow A \cup X_\bigcirc$
  $W_{\overline{\bigcirc}} \leftarrow \emptyset$
else
  $B \leftarrow Attr_{\overline{\bigcirc}}(G, X_{\overline{\bigcirc}})$
  $(Y_\Diamond, Y_\Box) \leftarrow Recursive(G \setminus B)$
  $W_\bigcirc \leftarrow Y_\bigcirc$
  $W_{\overline{\bigcirc}} \leftarrow B \cup Y_{\overline{\bigcirc}}$
end if
return $(W_\Diamond, W_\Box)$

Apply the recursive algorithm to the following parity game G




```
m ← 3
◯ ← □
U ← {v ∈ V | p(v) = 3} = {W, Z, Z'}
A ← Attr_□(G, U) = {W, Z, Z'}
(X_◇, X_□) ← Recursive(G \ V) = (∅, ∅)
if X_◇ = ∅ then
    W_□ ← A ∪ X_□ = A = {W, Z, Z'}
    W_◇ ← ∅
else
    ...
end if
return (W_◇, W_□) = (∅, {W, Z, Z'})
```

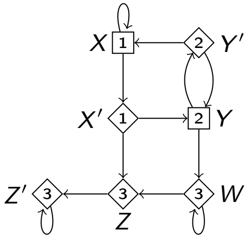
Apply the recursive algorithm to the following parity game G



$m \leftarrow 2$
$\bigcirc \leftarrow \Diamond$
$U \leftarrow \{v \in V \mid p(v) = 2\} = \{Y, Y'\}$
$A \leftarrow Attr_{\Diamond}(G, U) = \{Y, Y'\}$
$(X_{\Diamond}, X_{\Box}) \leftarrow Recursive(G \setminus \{Y, Y'\}) = (\emptyset, \{Z, Z', W\})$
if $X_{\Box} = \emptyset$ **then**
  $\dots$
else
  $B \leftarrow Attr_{\Box}(G, X_{\Box}) = \{Y, Y', Z, Z', W\}$
  $(Y_{\Diamond}, Y_{\Box}) \leftarrow Recursive(G \setminus V) = (\emptyset, \emptyset)$
  $W_{\Diamond} \leftarrow Y_{\Diamond} = \emptyset$
  $W_{\Box} \leftarrow B \cup Y_{\Box} = B = \{Y, Y', Z, Z', W\}$
end if
return $(W_{\Diamond}, W_{\Box}) = (\emptyset, \{Y, Y', Z, Z', W\})$

Example ($Recursive(G)$)

Consider parity game G:



$m \leftarrow 1$
$\bigcirc \leftarrow \Box$
$U \leftarrow \{v \in V \mid p(v) = 1\} = \{X, X'\}$
$A \leftarrow Attr_{\Box}(G, U) = \{X, X'\}$
$(X_{\Diamond}, X_{\Box}) \leftarrow Recursive(G \setminus \{X, X'\}) = (\emptyset, \{Y, Y', Z, Z', W\})$
if $X_{\Diamond} = \emptyset$ **then**
    $W_{\Box} \leftarrow A \cup X_{\Diamond} = \{X, X', Y, Y', Z, Z', W\}$
    $W_{\Diamond} \leftarrow \emptyset$
else
    ...
end if
return $(W_{\Diamond}, W_{\Box}) = (\emptyset, \{X, X', Y, Y', Z, Z', W\})$

So, player $\Box$ wins from all vertices!

Complexity

Let $G = (V, E, p, (V_\Diamond, V_\Box))$ be a parity game. $n = |V|, e = |E|, d = |\{p(v) \mid v \in V\}|$.

Worst-case running time complexity

$$\mathcal{O}(e \cdot n^{d})$$

Lowerbound on worst-case:

$$\Omega(fib(n)) = \Omega((\frac{1+\sqrt{5}}{2})^{n})$$

Technische Universiteit Eindhoven University of Technology

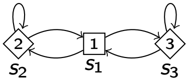
Complexity

Let $G = (V, E, p, (V_\Diamond, V_\Box))$ be a parity game; $n = |V|, e = |E|, d = |\{p(v) \mid v \in V\}|$.

- Algorithm with best known upper bound: Big step algorithm due to Schewe, with complexity
$$\mathcal{O}(d \cdot n^{d/3})$$
- Big step combines recursive algorithm with small progress measures;
- Small progress measures will be discussed first lecture in January

Exercise

Consider the following parity game:



- Compute the winning sets $W_\diamond$, $W_\Box$ for players $\diamond$ and $\Box$ in this parity game using the recursive algorithm.
- Translate this parity game to BES and solve the BES using Gauss elimination.