Formal Verification using Parity Games

Mathias N. Justesen

DTU Compute, Technical University of Denmark (DTU)

Background

- Many problems within formal verification can be reduced to solving parity games
  - Model checking (Stirling, 1995)
  - Controller synthesis (Arnold *et al.* , 2003)
  - Satisfiability (Friedmann & Lange, 2009b)

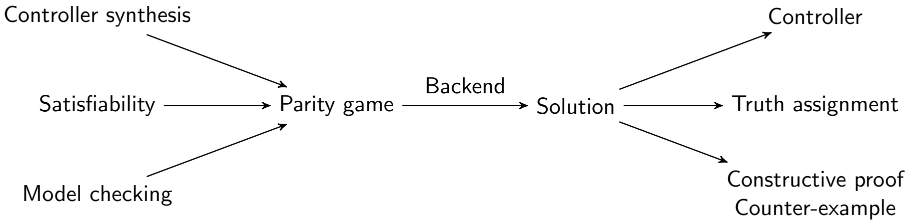
Background

- Many problems within formal verification can be reduced to solving parity games
  - Model checking (Stirling, 1995)
  - Controller synthesis (Arnold *et al.* , 2003)
  - Satisfiability (Friedmann & Lange, 2009b)
- Practical work restricted to model checking
  - mCRL2 and LTSmin
  - PBES to parity game

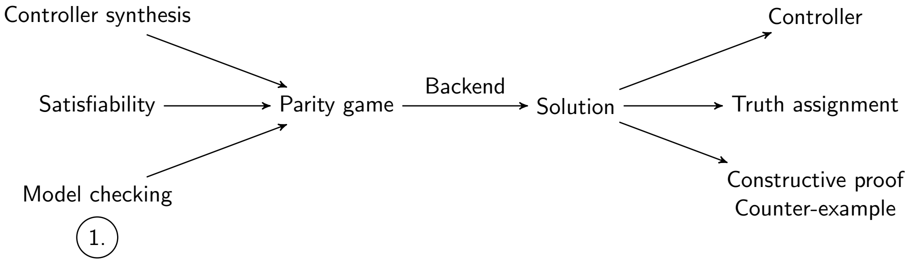
Background

- Many problems within formal verification can be reduced to solving parity games
  - Model checking (Stirling, 1995)
  - Controller synthesis (Arnold *et al.* , 2003)
  - Satisfiability (Friedmann & Lange, 2009b)
- Practical work restricted to model checking
  - mCRL2 and LTSmin
  - PBES to parity game
- Verification framework based on parity game solving

Framework

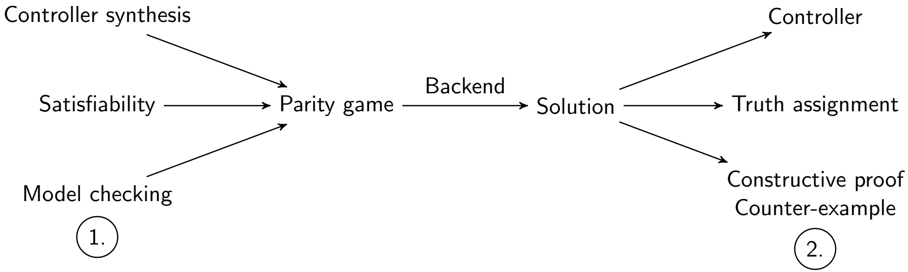

Controller synthesis
Satisfiability
Model checking
Parity game
Backend
Solution
Controller
Truth assignment
Constructive proof
Counter-example

Framework



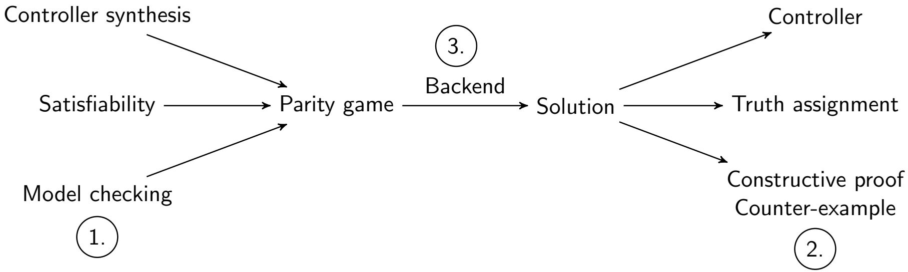
1. Model-checking for the modal μ-calculus
   - Semantics based on evaluation games
   - Conversion from evaluation game to parity game

Framework



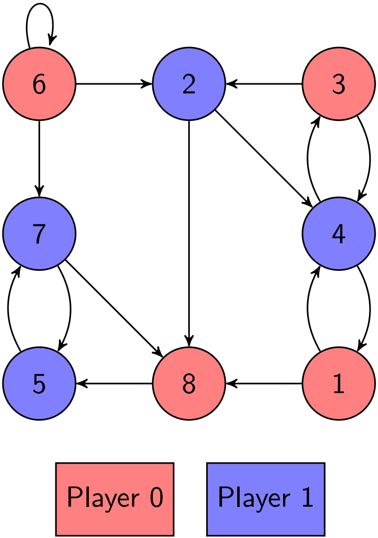
1. Model-checking for the modal μ-calculus
   - Semantics based on evaluation games
   - Conversion from evaluation game to parity game
2. Use solution to construct proof or counter-example

Framework



1. Model-checking for the modal μ-calculus
   - Semantics based on evaluation games
   - Conversion from evaluation game to parity game
2. Use solution to construct proof or counter-example
3. Backend based on PGSolver
   - Solve parity games in normal form

Parity Game

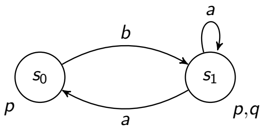


Model Checking

- $\mathbb{M} \models \varphi$?

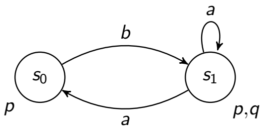
Model Checking

- $\mathbb{M} \models \varphi$?
- $\mathbb{M}$ is a Labelled Transition System



Model Checking

- $\mathbb{M} \models \varphi$?
- $\mathbb{M}$ is a Labelled Transition System

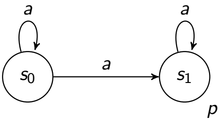


- Formulas of modal μ-calculus given proposition variables P and actions A:

$$\varphi ::= \top \mid \bot \mid p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \langle a \rangle \varphi \mid [a]\varphi \mid \mu x.\varphi \mid \nu x.\varphi$$

  where $p, x \in P$ and $a \in A$

Evaluation Game



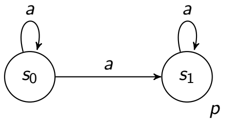
$\mu x.p \vee [a]x$

Player 0: Prove

Player 1: Disprove

Construction cf. (Venema, 2008)

Evaluation Game



$\mu x.p \vee [a]x, s_0$

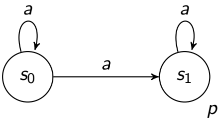
$\mu x.p \vee [a]x$

Player 0: Prove

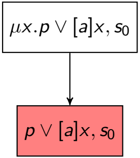
Player 1: Disprove

Construction cf. (Venema, 2008)

Evaluation Game



$\mu x.p \vee [a]x$

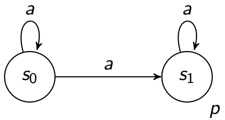


Player 0: Prove

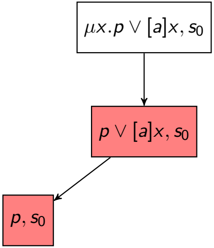
Player 1: Disprove

Construction cf. (Venema, 2008)

Evaluation Game



$\mu x.p \vee [a]x$

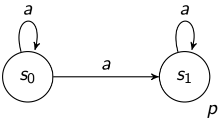


Player 0: Prove

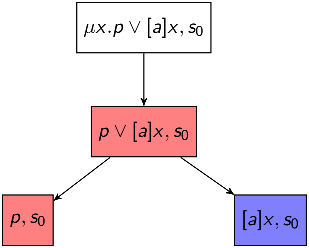
Player 1: Disprove

Construction cf. (Venema, 2008)

Evaluation Game



$\mu x.p \vee [a]x$

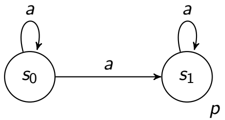


Player 0: Prove

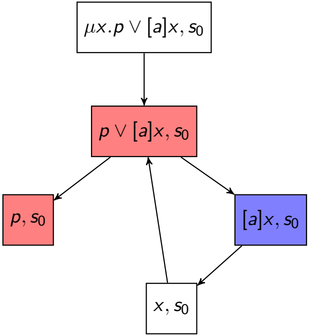
Player 1: Disprove

Construction cf. (Venema, 2008)

Evaluation Game



$\mu x.p \vee [a]x$

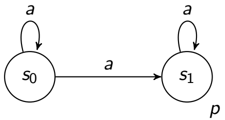


Player 0: Prove

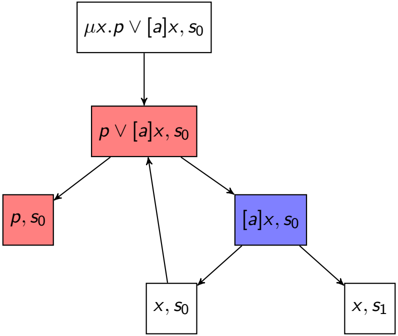
Player 1: Disprove

Construction cf. (Venema, 2008)

Evaluation Game



$\mu x.p \vee [a]x$

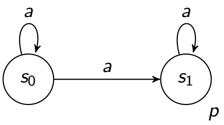


Player 0: Prove

Player 1: Disprove

Construction cf. (Venema, 2008)

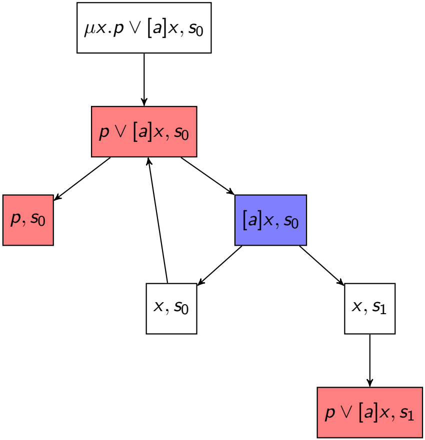
Evaluation Game



$\mu x.p \vee [a]x$

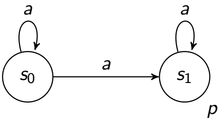
Player 0: Prove

Player 1: Disprove

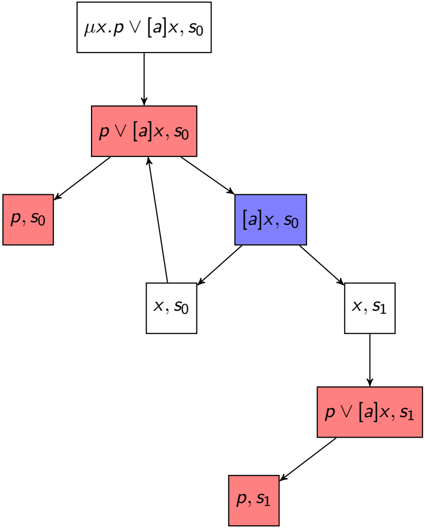


Construction cf. (Venema, 2008)

Evaluation Game

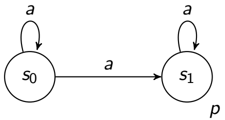


$\mu x.p \vee [a]x$



Construction cf. (Venema, 2008)

Evaluation Game

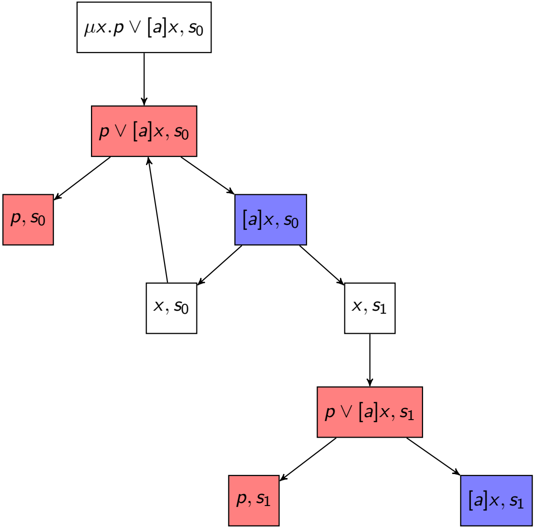


$\mu x.p \vee [a]x$

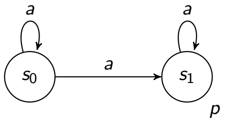
Player 0: Prove

Player 1: Disprove

Construction cf. (Venema, 2008)



Evaluation Game

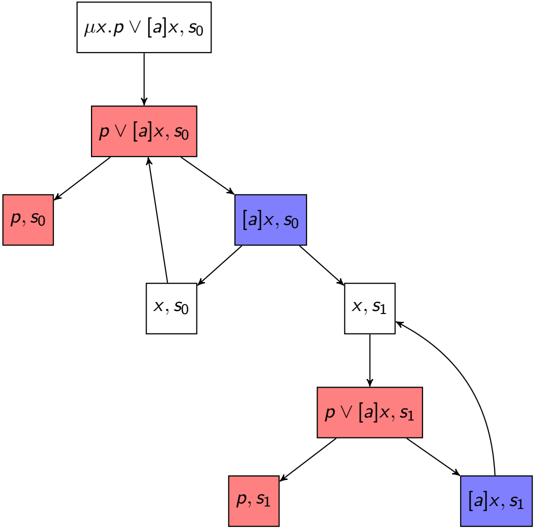


$\mu x.p \vee [a]x$

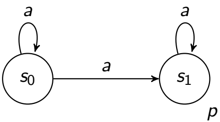
Player 0: Prove

Player 1: Disprove

Construction cf. (Venema, 2008)



Evaluation Game

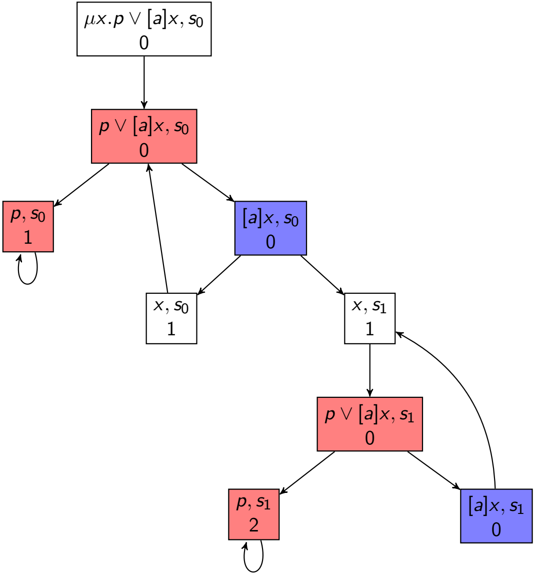


$\mu x.p \vee [a]x$

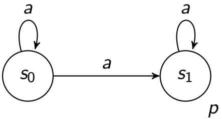
Player 0: Prove

Player 1: Disprove

Construction cf. (Venema, 2008)



Evaluation Game

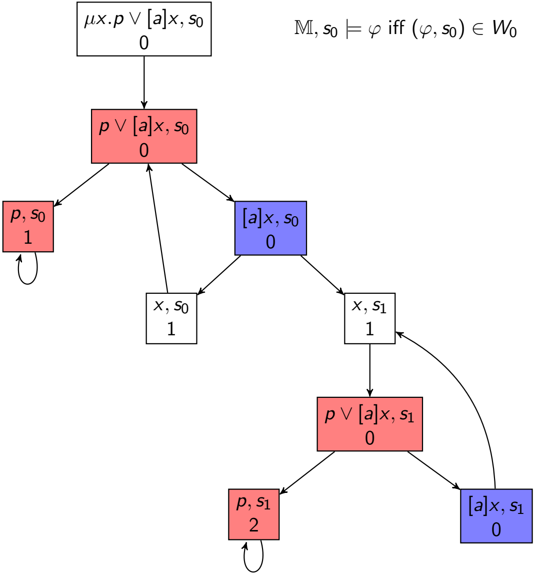


$\mu x.p \vee [a]x$

Player 0: Prove

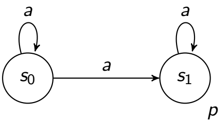
Player 1: Disprove

Construction cf. (Venema, 2008)



$\mathbb{M}, s_0 \models \varphi$ iff $(\varphi, s_0) \in W_0$

Evaluation Game

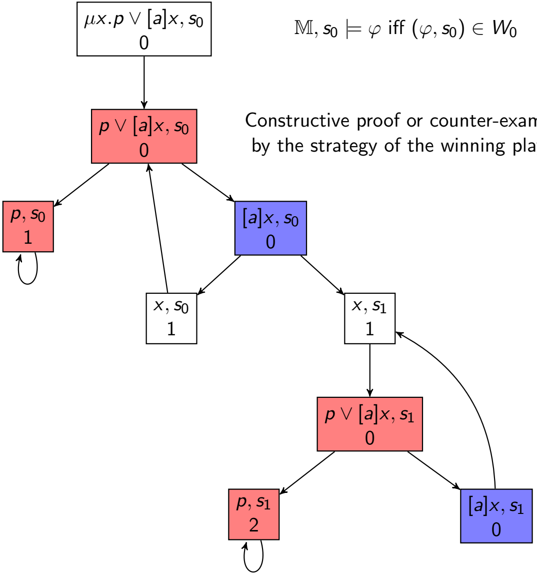


$\mu x.p \vee [a]x$

Player 0: Prove

Player 1: Disprove

Construction cf. (Venema, 2008)



$\mathbb{M}, s_0 \models \varphi$ iff $(\varphi, s_0) \in W_0$

Constructive proof or counter-example by the strategy of the winning player

Backend Solver

Backend Solver

- Dominion Decomposition Algorithm (Jurdzinski *et al.* , 2008)
    - Runtime: $O(n^{\sqrt{n}})$
    - Bad performance in practice

Backend Solver

- Dominion Decomposition Algorithm (Jurdzinski *et al.* , 2008)
  - Runtime: $O(n^{\sqrt{n}})$
  - Bad performance in practice
- Zielonka's Recursive Algorithm (Zielonka, 1998)
  - Runtime: $O(n^d)$
  - Good performance in practice (Friedmann & Lange, 2009a)

Backend Solver

- Dominion Decomposition Algorithm (Jurdzinski *et al.* , 2008)
    - Runtime: $O(n^{\sqrt{n}})$
    - Bad performance in practice
- Zielonka's Recursive Algorithm (Zielonka, 1998)
    - Runtime: $O(n^d)$
    - Good performance in practice (Friedmann & Lange, 2009a)
- Normal-Form Algorithm 1 (Vester, 2015)
    - Considers parity games in normal form

Backend Solver

- Dominion Decomposition Algorithm (Jurdzinski *et al.* , 2008)
  - Runtime: $O(n^{\sqrt{n}})$
  - Bad performance in practice
- Zielonka's Recursive Algorithm (Zielonka, 1998)
  - Runtime: $O(n^d)$
  - Good performance in practice (Friedmann & Lange, 2009a)
- Normal-Form Algorithm 1 (Vester, 2015)
  - Considers parity games in normal form
- Normal-Form Algorithm 2
  - Improved version of Normal-Form Algorithm 1

Definition

- A parity game in normal form if

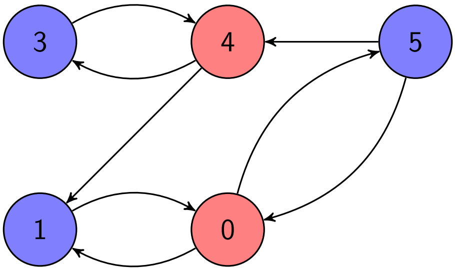
Definition

- A parity game in normal form if
  - It is truly turn-based,

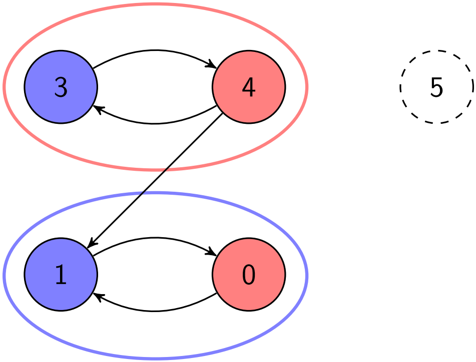
Definition

- A parity game in normal form if
  - It is truly turn-based,
  - Player 0 controls only nodes of even priority, and
  - Player 1 controls only nodes of odd priority

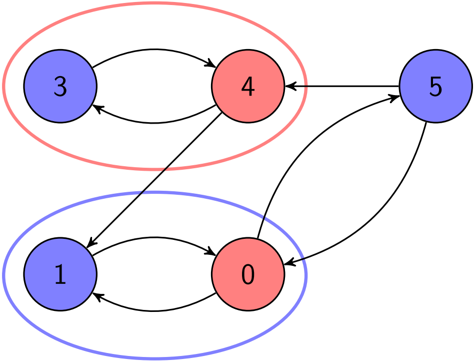
Example



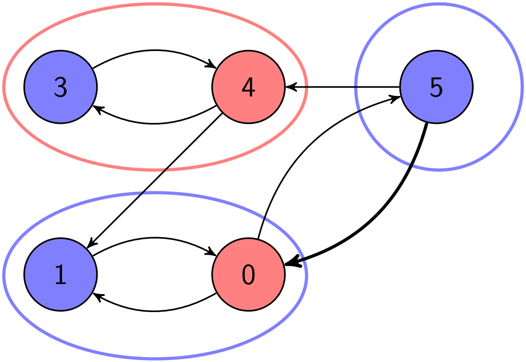
Example



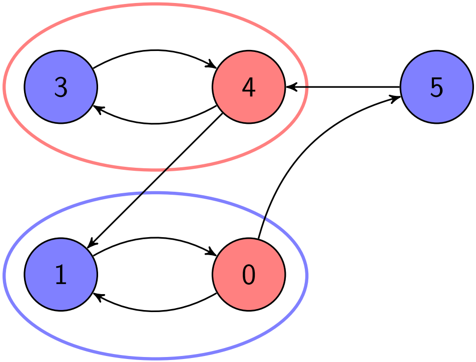
Example



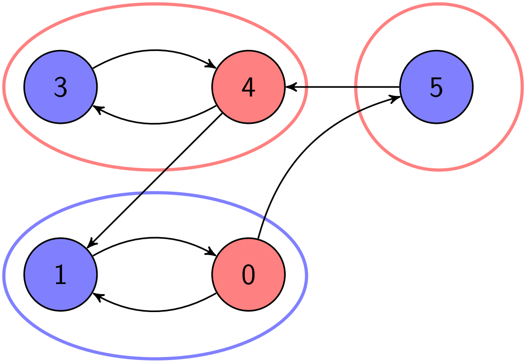
Example



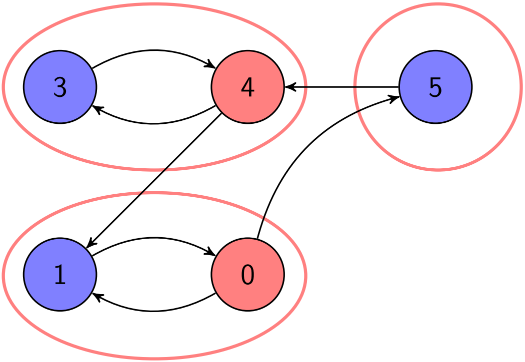
Example



Example



Example



Advantages and Disadvantages

- Quickly decide if a node is winning for Player 0 or Player 1

Advantages and Disadvantages

- Quickly decide if a node is winning for Player 0 or Player 1
- Many recursive calls - one per node

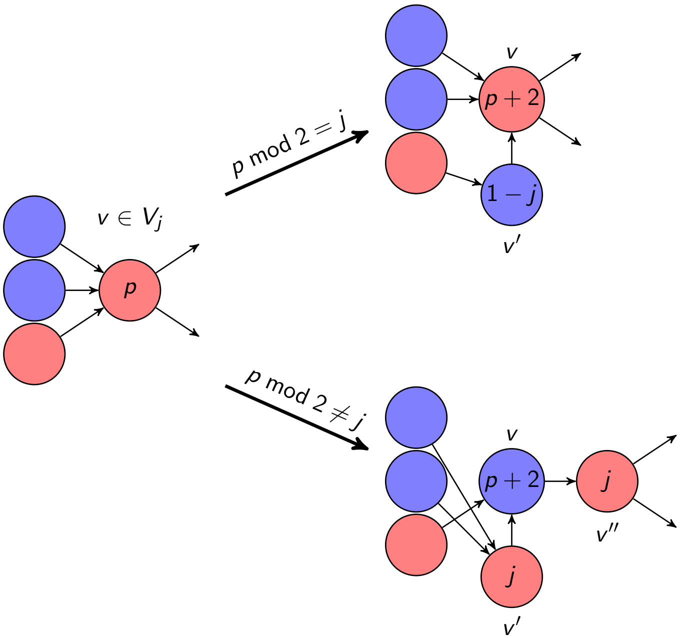
Advantages and Disadvantages

- Quickly decide if a node is winning for Player 0 or Player 1
- Many recursive calls - one per node
- Normal-Form Algorithm 2 addresses this issue by considering all nodes of the same priority at the same time

Advantages and Disadvantages

- Quickly decide if a node is winning for Player 0 or Player 1
- Many recursive calls - one per node
- Normal-Form Algorithm 2 addresses this issue by considering all nodes of the same priority at the same time
- Algorithms restricted to games in normal form

Transformation



Comparison of Algorithms

$n, d, deg_{min}, deg_{max}$	Not NF			Pre-NF			NF		
	Zie	NF1	NF2	Zie	NF1	NF2	Zie	NF1	NF2
100, 100, 2, 4	0.00	10.55	0.42	0.00	10.58	0.41	0.00	0.04	0.02
100, 100, 2, 10	0.00	6.13	0.29	0.00	6.16	0.28	0.00	0.01	0.01
100, 100, 2, 100	0.00	3.47	0.18	0.00	3.45	0.19	**0.01**	**0.01**	**0.01**
200, 200, 2, 4	0.00		11.01	0.00		10.78	0.01	0.43	0.23
200, 200, 2, 10	0.00		2.37	0.00		2.29	0.01	0.22	0.16
200, 200, 2, 200	0.01	69.29	2.29	0.01	52.05	2.27	**0.05**	**0.05**	**0.03**
500, 500, 2, 4	0.00			0.01			0.07		
500, 500, 2, 10	0.01			0.03			0.10	13.24	6.31
500, 500, 2, 500	0.07		78.01	0.08		77.18	**1.11**	**1.04**	**0.73**
Rec. ladder 5	0.00	0.03	0.01						
Rec. ladder 10	0.01	5.94	0.75						
Rec. ladder 15	0.07		94.36						

Testing the Limits

$$\varphi_n = \psi_n \vee \neg\psi_n$$

Testing the Limits

$$\varphi_n = \psi_n \vee \neg\psi_n$$

$$\psi_n = \mu x_1.\nu x_2 \ldots \eta_n x_n.\Big(q_1 \vee \langle\ \rangle\big(x_1 \wedge (q_2 \vee \langle\ \rangle(x_1 \wedge \ldots (q_n \vee \langle\ \rangle x_n)))\big)\Big)$$

Testing the Limits

$$\varphi_n = \psi_n \vee \neg\psi_n$$

$$\psi_n = \mu x_1.\nu x_2 \ldots \eta_n x_n.\Big(q_1 \vee \langle\ \rangle\big(x_1 \wedge (q_2 \vee \langle\ \rangle(x_1 \wedge \ldots (q_n \vee \langle\ \rangle x_n)))\big)\Big)$$

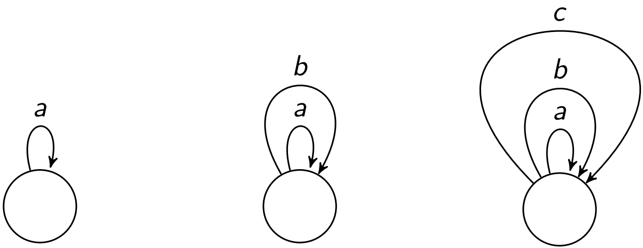
$$\langle\ \rangle\varphi = \bigvee_{a \in A} \langle a \rangle \varphi$$

Testing the Limits

$$\varphi_n = \psi_n \vee \neg\psi_n$$

$$\psi_n = \mu x_1.\nu x_2 \ldots \eta_n x_n.\Big(q_1 \vee \langle\ \rangle\big(x_1 \wedge (q_2 \vee \langle\ \rangle(x_1 \wedge \ldots (q_n \vee \langle\ \rangle x_n)))\big)\Big)$$

$$\langle\ \rangle\varphi = \bigvee_{a\in A} \langle a\rangle\varphi$$



(a) $\mathbb{L}_1$ (b) $\mathbb{L}_2$ (c) $\mathbb{L}_3$

Testing the Limits

LTS	Nodes	n	Time
$\mathbb{L}_1$	12.000	1024	3:27.4
$\mathbb{L}_2$	786.000	16	0:03.6
$\mathbb{L}_2$	1.573.000	17	0:03.8
$\mathbb{L}_3$	413.000	10	0:01.8
$\mathbb{L}_3$	1.240.000	11	0:05.6
$\mathbb{L}_3$	3.720.000	12	0:07.6

Testing the Limits

LTS	Nodes	n	Time
$\mathbb{L}_1$	12.000	1024	3:27.4
$\mathbb{L}_2$	786.000	16	0:03.6
$\mathbb{L}_2$	1.573.000	17	0:03.8
$\mathbb{L}_3$	413.000	10	0:01.8
$\mathbb{L}_3$	1.240.000	11	0:05.6
$\mathbb{L}_3$	3.720.000	12	0:07.6

State space: $O\big(|\mathbb{M}| \cdot |Sfor(\varphi)|\big)$

Conclusions

- Parity game solving is well suited for model checking

Conclusions

- Parity game solving is well suited for model checking
- Zielonka's Algorithm works well in practice

Conclusions

- Parity game solving is well suited for model checking
- Zielonka's Algorithm works well in practice
- Future work

Conclusions

- Parity game solving is well suited for model checking
- Zielonka's Algorithm works well in practice
- Future work
  - Specialized algorithms

Conclusions

- Parity game solving is well suited for model checking
- Zielonka's Algorithm works well in practice
- Future work
  - Specialized algorithms
  - Winning cores

Conclusions

- Parity game solving is well suited for model checking
- Zielonka's Algorithm works well in practice
- Future work
  - Specialized algorithms
  - Winning cores
  - Controller synthesis (Arnold *et al.* , 2003; Ramadge & Wonham, 1989)

Conclusions

- Parity game solving is well suited for model checking
- Zielonka's Algorithm works well in practice
- Future work
  - Specialized algorithms
  - Winning cores
  - Controller synthesis (Arnold *et al.* , 2003; Ramadge & Wonham, 1989)
  - Symbolic representation of parity games (Kant & van de Pol, 2014)

References I

Arnold, A., Vincent, A., & Walukiewicz, I. 2003. Games for synthesis of controllers with partial observation. *Theoretical Computer Science*, **303**(1), 7 – 34. Logic and Complexity in Computer Science.

Artale, Alessandro. 2011. *Formal Methods — Lecture III: Linear Temporal Logic*. URL: https://www.inf.unibz.it/~artale/FM/slide3.pdf.

Friedmann, Oliver, & Lange, Martin. 2009a. Solving Parity Games in Practice. *Pages 182–196 of:* Liu, Zhiming, & Ravn, Anders P. (eds), *Automated Technology for Verification and Analysis*. Lecture Notes in Computer Science, vol. 5799. Springer Berlin Heidelberg.

Friedmann, Oliver, & Lange, Martin. 2009b. Tableaux with automata. *In: Proc. Workshop on Tableaux vs. Automata as Logical Decision Procedures, AutoTab*, vol. 9.

Jurdzinski, Marcin, Paterson, Mike, & Zwick, Uri. 2008. A deterministic subexponential algorithm for solving parity games. *SIAM Journal on Computing*, **38**(4), 1519–1532.

References II

Kant, Gijs, & van de Pol, Jaco. 2014. Generating and Solving Symbolic Parity Games. *Pages 2–14 of: Proceedings 3rd Workshop on GRAPH Inspection and Traversal Engineering, GRAPHITE 2014, Grenoble, France, 5th April 2014.*

Ramadge, P.J.G., & Wonham, W.M. 1989. The control of discrete event systems. *Proceedings of the IEEE*, **77**(1), 81–98.

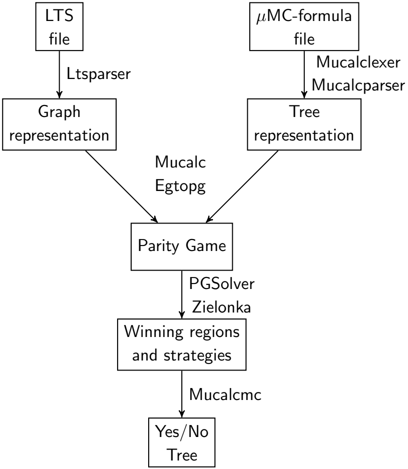
Stirling, Colin. 1995. Local model checking games. *Pages 1–11 of: CONCUR'95: Concurrency Theory*. Springer.

Venema, Yde. 2008. Lectures on the modal μ-calculus. *Institute for Logic, Language and Computation, University of Amsterdam*.

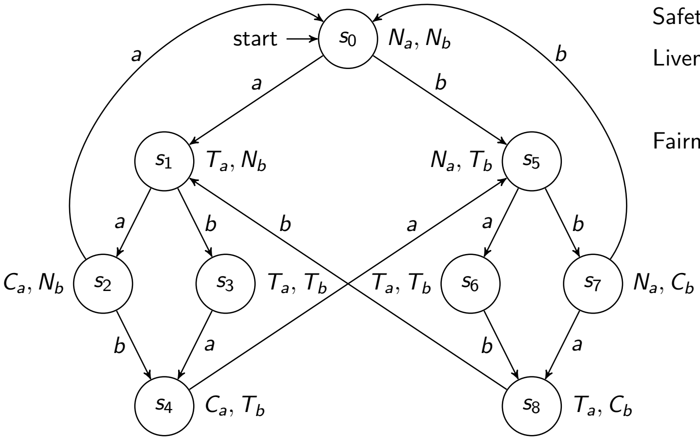
Vester, Steen. 2015. *A New Algorithm for Solving Parity Games*.

Zielonka, Wieslaw. 1998. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science*, **200**(1–2), 135 – 183.

Implementation



Mutual Exclusion



Safety: $\Box\neg(C_a \wedge C_b)$

Liveness: $\Diamond C_a$

$\Box(T_a \rightarrow \Diamond C_a)$

Fairness: $\Box\Diamond C_a$

$\Box\Diamond T_a \rightarrow \Box\Diamond C_a$

Example from (Artale, 2011)