Solving parity games

Madhavan Mukund

Chennai Mathematical Institute
http://www.cmi.ac.in/~madhavan

Formal Methods Update 2006, IIT Guwahati
4 July 2006

Outline

- Parity games
- An efficient algorithm for solving parity games [Jurdziński]
- Solving parity games through strategy improvement [Jurdziński and Vöge]

Parity games

- Two players, 0 and 1

Parity games

- Two players, 0 and 1
- Game graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, $\mathbf{V} = \mathbf{V}_0 \uplus \mathbf{V}_1$

Parity games

- Two players, 0 and 1
- Game graph $\mathbf{G = (V, E)}$, $\mathbf{V = V_0 \uplus V_1}$
  - Player 0 plays from $\mathbf{V_0}$, player 1 from $\mathbf{V_1}$

Parity games

- Two players, 0 and 1
- Game graph $\mathbf{G = (V, E)}$, $\mathbf{V = V_0 \uplus V_1}$
  - Player 0 plays from $\mathbf{V_0}$, player 1 from $\mathbf{V_1}$
  - From every position, at least one move is possible

Parity games

- Two players, 0 and 1
- Game graph $\mathbf{G = (V, E)}$, $\mathbf{V = V_0 \uplus V_1}$
  - Player 0 plays from $\mathbf{V_0}$, player 1 from $\mathbf{V_1}$
  - From every position, at least one move is possible
- $\mathbf{c : V \longrightarrow \mathbb{N}}$ assigns a colour to each position

Parity games

- Two players, 0 and 1
- Game graph $\mathbf{G = (V, E)}$, $\mathbf{V = V_0 \uplus V_1}$
  - Player 0 plays from $\mathbf{V_0}$, player 1 from $\mathbf{V_1}$
  - From every position, at least one move is possible
- $\mathbf{c : V \longrightarrow \mathbb{N}}$ assigns a colour to each position
- Player 0 wins an infinite play if it satifies the parity winning condition

Parity games

- Two players, 0 and 1
- Game graph $\mathbf{G = (V, E)}$, $\mathbf{V = V_0 \uplus V_1}$
  - Player 0 plays from $\mathbf{V_0}$, player 1 from $\mathbf{V_1}$
  - From every position, at least one move is possible
- $\mathbf{c : V \longrightarrow \mathbb{N}}$ assigns a colour to each position
- Player 0 wins an infinite play if it satifies the parity winning condition
  - Max-parity: largest colour that occurs infinitely often in the play is even

Parity games

- Two players, 0 and 1
- Game graph $\mathbf{G = (V, E)}$, $\mathbf{V = V_0 \uplus V_1}$
  - Player 0 plays from $\mathbf{V_0}$, player 1 from $\mathbf{V_1}$
  - From every position, at least one move is possible
- $\mathbf{c : V \longrightarrow \mathbb{N}}$ assigns a colour to each position
- Player 0 wins an infinite play if it satifies the parity winning condition
  - Max-parity: largest colour that occurs infinitely often in the play is even
  - Min-parity: smallest colour that occurs infinitely often in the play is even

Memoryless determinacy for parity games

Theorem

The set of positions of a parity game can be partitioned as W_0, from where player 0 wins with a memoryless strategy, and W_1, from where player 1 wins with a memoryless strategy.

- Can identify W_0 and W_1 recursively, using 0-paradises and 1-paradises

Memoryless determinacy for parity games

Theorem

The set of positions of a parity game can be partitioned as $\mathbf{W_0}$, from where player 0 wins with a memoryless strategy, and $\mathbf{W_1}$, from where player 1 wins with a memoryless strategy.

- Can identify $\mathbf{W_0}$ and $\mathbf{W_1}$ recursively, using 0-paradises and 1-paradises
- Complexity is $\mathbf{O(mn^d)}$

Memoryless determinacy for parity games

Theorem

The set of positions of a parity game can be partitioned as $\mathbf{W_0}$, from where player 0 wins with a memoryless strategy, and $\mathbf{W_1}$, from where player 1 wins with a memoryless strategy.

- Can identify $\mathbf{W_0}$ and $\mathbf{W_1}$ recursively, using 0-paradises and 1-paradises
- Complexity is $\mathbf{O(mn^d)}$
  - $\mathbf{m}$ edges, $\mathbf{n}$ states, largest colour $\mathbf{d}$

Memoryless determinacy for parity games

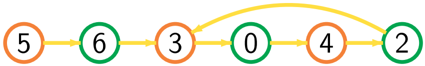
Theorem

The set of positions of a parity game can be partitioned as $\mathbf{W_0}$, from where player 0 wins with a memoryless strategy, and $\mathbf{W_1}$, from where player 1 wins with a memoryless strategy.

- Can identify $\mathbf{W_0}$ and $\mathbf{W_1}$ recursively, using 0-paradises and 1-paradises
- Complexity is $\mathbf{O(mn^d)}$
  - $\mathbf{m}$ edges, $\mathbf{n}$ states, largest colour $\mathbf{d}$
- Can we identify $\mathbf{W_0}$ and $\mathbf{W_1}$ more efficiently?

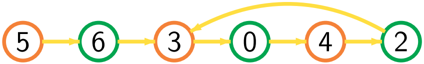
Solitaire games

- Observation If both players play by memoryless strategy, each infinite play is a finite prefix followed by a simple loop



Solitaire games

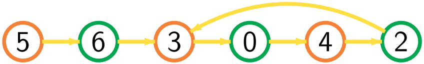
- Observation If both players play by memoryless strategy, each infinite play is a finite prefix followed by a simple loop



- Let f_0 be a strategy for Player 0

Solitaire games

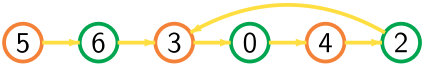
- Observation If both players play by memoryless strategy, each infinite play is a finite prefix followed by a simple loop



- Let f_0 be a strategy for Player 0
- f_0 is closed for a set of positions X if all plays that start in X that are consistent with f_0 stay in X

Solitaire games

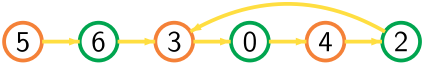
- Observation If both players play by memoryless strategy, each infinite play is a finite prefix followed by a simple loop



- Let f_0 be a strategy for Player 0
- f_0 is closed for a set of positions X if all plays that start in X that are consistent with f_0 stay in X
- Remove all moves not consistent with f_0 to get a solitaire game for Player 1

Solitaire games

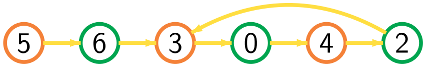
- Observation If both players play by memoryless strategy, each infinite play is a finite prefix followed by a simple loop



- Let f_0 be a strategy for Player 0
- f_0 is closed for a set of positions X if all plays that start in X that are consistent with f_0 stay in X
- Remove all moves not consistent with f_0 to get a solitaire game for Player 1
- Odd/even cycle—simple cycle in solitaire game with minimum colour odd/even

Solitaire games

- Observation If both players play by memoryless strategy, each infinite play is a finite prefix followed by a simple loop



- Let f_0 be a strategy for Player 0
- f_0 is closed for a set of positions X if all plays that start in X that are consistent with f_0 stay in X
- Remove all moves not consistent with f_0 to get a solitaire game for Player 1
- Odd/even cycle—simple cycle in solitaire game with minimum colour odd/even

Lemma

f_0 closed on X wins from all states in X iff all simple cycles in the game restricted to X are even.

Parity progress measures

- For a game with $\mathbf{d}$ colours, assign a $\mathbf{d+1}$-tuple $\rho(\mathbf{v}) = (\mathbf{n_0}, \mathbf{n_1}, \ldots, \mathbf{n_d})$ to each position

Parity progress measures

- For a game with $\mathbf{d}$ colours, assign a $\mathbf{d+1}$-tuple $\rho(\mathbf{v}) = (\mathbf{n_0}, \mathbf{n_1}, \ldots, \mathbf{n_d})$ to each position
  - Compare $\mathbf{d}$-tuples lexicographically
  - $(\mathbf{x_0}, \ldots, \mathbf{x_d}) \geq_{\mathbf{i}} (\mathbf{y_0}, \ldots, \mathbf{y_d})$: lexicographic comparison using first $\mathbf{i}$ components

Parity progress measures

- For a game with d colours, assign a $d+1$-tuple $\rho(v) = (n_0, n_1, \ldots, n_d)$ to each position
  - Compare d-tuples lexicographically
  - $(x_0, \ldots, x_d) \geq_i (y_0, \ldots, y_d)$: lexicographic comparison using first i components
- Parity progress measure
  For each edge $v \longrightarrow w$
  - $c(v)$ even $\Rightarrow \rho(v) \geq_{c(v)} \rho(w)$
  - $c(v)$ odd $\Rightarrow \rho(v) >_{c(v)} \rho(w)$

Parity progress measures

- For a game with $\mathbf{d}$ colours, assign a $\mathbf{d+1}$-tuple $\rho(\mathbf{v}) = (\mathbf{n_0}, \mathbf{n_1}, \ldots, \mathbf{n_d})$ to each position
  - Compare $\mathbf{d}$-tuples lexicographically
  - $(\mathbf{x_0}, \ldots, \mathbf{x_d}) \geq_{\mathbf{i}} (\mathbf{y_0}, \ldots, \mathbf{y_d})$: lexicographic comparison using first $\mathbf{i}$ components
- Parity progress measure
  For each edge $\mathbf{v} \longrightarrow \mathbf{w}$
  - $\mathbf{c(v)}$ even $\Rightarrow \rho(\mathbf{v}) \geq_{\mathbf{c(v)}} \rho(\mathbf{w})$
  - $\mathbf{c(v)}$ odd $\Rightarrow \rho(\mathbf{v}) >_{\mathbf{c(v)}} \rho(\mathbf{w})$

Lemma

If a solitaire game admits a parity progress measure, then every simple cycle in the game is even.

Parity progress measures . . .

Lemma

If every simple cycle in a solitaire game is even, we can construct a small parity progress measure.

Parity progress measures ...

Lemma

If every simple cycle in a solitaire game is even, we can construct a small parity progress measure.

- Construct $\rho : \mathbf{v} \mapsto (\mathbf{n_0}, \mathbf{n_1}, \ldots, \mathbf{n_d})$ (assume that $\mathbf{d}$ is odd)

Parity progress measures . . .

Lemma

If every simple cycle in a solitaire game is even, we can construct a small parity progress measure.

- Construct $\rho : v \mapsto (n_0, n_1, \ldots, n_d)$ (assume that d is odd)
  - For each even i, $n_i = 0$

Parity progress measures . . .

Lemma

If every simple cycle in a solitaire game is even, we can construct a small parity progress measure.

- Construct $\rho : v \mapsto (n_0, n_1, \ldots, n_d)$ (assume that d is odd)
  - For each even i, $n_i = 0$
  - For each odd i, define n_i as follows:

Parity progress measures . . .

Lemma

If every simple cycle in a solitaire game is even, we can construct a small parity progress measure.

- Construct $\rho : v \mapsto (n_0, n_1, \ldots, n_d)$ (assume that d is odd)
  - For each even i, $n_i = 0$
  - For each odd i, define n_i as follows:

    Consider all infinite paths from v with minimum colour i. Set n_i to maximum number of times i appears along all such paths.

Parity progress measures . . .

Lemma

If every simple cycle in a solitaire game is even, we can construct a small parity progress measure.

- Construct $\rho : v \mapsto (n_0, n_1, \ldots, n_d)$ (assume that d is odd)
  - For each even i, $n_i = 0$
  - For each odd i, define n_i as follows:

    Consider all infinite paths from v with minimum colour i. Set n_i to maximum number of times i appears along all such paths.

    n_i may be set to 0 or ∞!

Parity progress measures . . .

Lemma

If every simple cycle in a solitaire game is even, we can construct a small parity progress measure.

- Construct $\rho : v \mapsto (n_0, n_1, \ldots, n_d)$ (assume that d is odd)
  - For each even i, $n_i = 0$
  - For each odd i, define n_i as follows:

    Consider all infinite paths from v with minimum colour i. Set n_i to maximum number of times i appears along all such paths.

    n_i may be set to 0 or ∞!
- Let V_i be set of positions coloured i

  Claim 1 Let $\rho(v) = (n_0, n_1, \ldots, n_d)$.
  For odd i, $n_i \leq |V_i + 1|$.

Parity progress measures . . .

Lemma

If every simple cycle in a solitaire game is even, we can construct a small parity progress measure.

- Construct $\rho : v \mapsto (n_0, n_1, \ldots, n_d)$ (assume that d is odd)
  - For each even i, $n_i = 0$
  - For each odd i, define n_i as follows:

    Consider all infinite paths from v with minimum colour i. Set n_i to maximum number of times i appears along all such paths.

    n_i may be set to 0 or ∞!
- Let V_i be set of positions coloured i

  Claim 1 Let $\rho(v) = (n_0, n_1, \ldots, n_d)$.
  For odd i, $n_i \leq |V_i + 1|$.
- **Claim 2** $\rho(v)$ is a parity progress measure.

Parity progress measures . . .

Lemma

If every simple cycle in a solitaire game is even, we can construct a small parity progress measure.

Parity progress measures . . .

Lemma

If every simple cycle in a solitaire game is even, we can construct a small parity progress measure.

- We have $\rho : v \mapsto (n_0, n_1, \ldots, n_d)$ such that $n_0 = n_2 = \cdots = n_{d-1} = 0$ and, for odd i, $n_i \leq |V_i|$ (recall that we assume d is odd)

Parity progress measures . . .

Lemma

If every simple cycle in a solitaire game is even, we can construct a small parity progress measure.

- We have $\rho : v \mapsto (n_0, n_1, \ldots, n_d)$ such that $n_0 = n_2 = \cdots = n_{d-1} = 0$ and, for odd i, $n_i \leq |V_i|$ (recall that we assume d is odd)
- Range of ρ is M where $M = \{0\} \times \{0, \ldots, |V_1|\} \times \{0\} \times \cdots \times \{0, \ldots, |V_d|\}$

Game progress measures

- From parity progress measures on solitaire games to game progress measures on full game graph

Game progress measures

- From parity progress measures on solitaire games to game progress measures on full game graph
- Extend
  $\mathbf{M} = \{0\} \times \{0, \ldots, |\mathbf{V_1}|\} \times \{0\} \times \cdots \times \{0, \ldots, |\mathbf{V_d}|\}$
  by adding a new element $\top$ bigger than all elements in $\mathbf{M}$

Game progress measures

- From parity progress measures on solitaire games to game progress measures on full game graph
- Extend
  $\mathbf{M} = \{0\} \times \{0, \ldots, |\mathbf{V_1}|\} \times \{0\} \times \cdots \times \{0, \ldots, |\mathbf{V_d}|\}$
  by adding a new element $\top$ bigger than all elements in $\mathbf{M}$
- Construct $\rho : \mathbf{v} \mapsto \mathbf{M}_\top$ so that

Game progress measures

- From parity progress measures on solitaire games to game progress measures on full game graph
- Extend
  $\mathbf{M} = \{0\} \times \{0, \ldots, |\mathbf{V_1}|\} \times \{0\} \times \cdots \times \{0, \ldots, |\mathbf{V_d}|\}$
  by adding a new element $\top$ bigger than all elements in $\mathbf{M}$
- Construct $\rho : \mathbf{v} \mapsto \mathbf{M}_\top$ so that
  - If $\mathbf{v} \in \mathbf{V_0}$, for some $\mathbf{v} \longrightarrow \mathbf{w}$, $\rho(\mathbf{v}) \geq_{\mathbf{c(v)}} \rho(\mathbf{w})$

Game progress measures

- From parity progress measures on solitaire games to game progress measures on full game graph
- Extend
  $\mathbf{M} = \{0\} \times \{0, \ldots, |\mathbf{V_1}|\} \times \{0\} \times \cdots \times \{0, \ldots, |\mathbf{V_d}|\}$
  by adding a new element $\top$ bigger than all elements in $\mathbf{M}$
- Construct $\rho : \mathbf{v} \mapsto \mathbf{M}_\top$ so that
  - If $\mathbf{v} \in \mathbf{V_0}$, for some $\mathbf{v} \longrightarrow \mathbf{w}$, $\rho(\mathbf{v}) \geq_{\mathbf{c(v)}} \rho(\mathbf{w})$
  - If $\mathbf{v} \in \mathbf{V_1}$, for every $\mathbf{v} \longrightarrow \mathbf{w}$, $\rho(\mathbf{v}) >_{\mathbf{c(v)}} \rho(\mathbf{w})$,
    unless $\rho(\mathbf{v}) = \rho(\mathbf{w}) = \top$

Game progress measures

- From parity progress measures on solitaire games to game progress measures on full game graph
- Extend
  $\mathbf{M} = \{0\} \times \{0, \ldots, |\mathbf{V_1}|\} \times \{0\} \times \cdots \times \{0, \ldots, |\mathbf{V_d}|\}$
  by adding a new element $\top$ bigger than all elements in $\mathbf{M}$
- Construct $\rho : \mathbf{v} \mapsto \mathbf{M}_\top$ so that
  - If $\mathbf{v} \in \mathbf{V_0}$, for some $\mathbf{v} \longrightarrow \mathbf{w}$, $\rho(\mathbf{v}) \geq_{\mathbf{c(v)}} \rho(\mathbf{w})$
  - If $\mathbf{v} \in \mathbf{V_1}$, for every $\mathbf{v} \longrightarrow \mathbf{w}$, $\rho(\mathbf{v}) >_{\mathbf{c(v)}} \rho(\mathbf{w})$,
    unless $\rho(\mathbf{v}) = \rho(\mathbf{w}) = \top$
- A trivial game progress measure assigns $\top$ everywhere.

Game progress measures

- From parity progress measures on solitaire games to game progress measures on full game graph
- Extend
  $\mathbf{M} = \{0\} \times \{0, \ldots, |\mathbf{V_1}|\} \times \{0\} \times \cdots \times \{0, \ldots, |\mathbf{V_d}|\}$
  by adding a new element $\top$ bigger than all elements in $\mathbf{M}$
- Construct $\rho : \mathbf{v} \mapsto \mathbf{M}_\top$ so that
  - If $\mathbf{v} \in \mathbf{V_0}$, for some $\mathbf{v} \longrightarrow \mathbf{w}$, $\rho(\mathbf{v}) \geq_{\mathbf{c(v)}} \rho(\mathbf{w})$
  - If $\mathbf{v} \in \mathbf{V_1}$, for every $\mathbf{v} \longrightarrow \mathbf{w}$, $\rho(\mathbf{v}) >_{\mathbf{c(v)}} \rho(\mathbf{w})$,
    unless $\rho(\mathbf{v}) = \rho(\mathbf{w}) = \top$
- A trivial game progress measure assigns $\top$ everywhere.
- Let $\|\rho\| = \{\mathbf{v} \mid \rho(\mathbf{v}) \neq \top\}$

Game progress measures

- From parity progress measures on solitaire games to game progress measures on full game graph
- Extend
  $\mathbf{M} = \{0\} \times \{0, \ldots, |\mathbf{V_1}|\} \times \{0\} \times \cdots \times \{0, \ldots, |\mathbf{V_d}|\}$
  by adding a new element $\top$ bigger than all elements in $\mathbf{M}$
- Construct $\rho : \mathbf{v} \mapsto \mathbf{M}_\top$ so that
  - If $\mathbf{v} \in \mathbf{V_0}$, for some $\mathbf{v} \longrightarrow \mathbf{w}$, $\rho(\mathbf{v}) \geq_{\mathbf{c(v)}} \rho(\mathbf{w})$
  - If $\mathbf{v} \in \mathbf{V_1}$, for every $\mathbf{v} \longrightarrow \mathbf{w}$, $\rho(\mathbf{v}) >_{\mathbf{c(v)}} \rho(\mathbf{w})$,
    unless $\rho(\mathbf{v}) = \rho(\mathbf{w}) = \top$
- A trivial game progress measure assigns $\top$ everywhere.
- Let $\|\rho\| = \{\mathbf{v} \mid \rho(\mathbf{v}) \neq \top\}$
- Our aim is to find ρ such that $\|\rho\|$ is maximized.

Game progress measures . . .

- Given ρ, define the strategy $\mathbf{f}_0^{\rho}$ that chooses for each position $\mathbf{v}$ the successor $\mathbf{w}$ with minimum $\rho(\mathbf{w})$

Game progress measures . . .

- Given ρ, define the strategy $\mathbf{f}_0^{\rho}$ that chooses for each position $\mathbf{v}$ the successor $\mathbf{w}$ with minimum $\rho(\mathbf{w})$

Lemma

$\mathbf{f}_0^{\rho}$ wins in the subgame defined by $||\rho||$

Game progress measures . . .

- Given ρ, define the strategy $\mathbf{f}_0^{\rho}$ that chooses for each position $\mathbf{v}$ the successor $\mathbf{w}$ with minimum $\rho(\mathbf{w})$

Lemma

$\mathbf{f}_0^{\rho}$ wins in the subgame defined by $||\rho||$

Lemma

There is a game progress measure ρ such that $||\rho||$ is the winning region for Player 0.

Game progress measures . . .

- Given ρ, define the strategy $\mathbf{f}_0^{\rho}$ that chooses for each position $\mathbf{v}$ the successor $\mathbf{w}$ with minimum $\rho(\mathbf{w})$

Lemma

$\mathbf{f}_0^{\rho}$ wins in the subgame defined by $\|\rho\|$

Lemma

There is a game progress measure ρ such that $\|\rho\|$ is the winning region for Player 0.

- Player 0 has a memoryless winning strategy $\mathbf{f}_0$ with winning set $\mathbf{W}_0$. The solitaire game over $\mathbf{W}_0$ defined by $\mathbf{f}_0$ has only even cycles $\Rightarrow$ we can assign a parity progress measure over $\mathbf{W}_0$, which lifts to a game progress measure ρ with $\mathbf{W}_0 = \|\rho\|$.

Computing game progress measures

- Define an operator $\mathbf{Lift}(\rho, \mathbf{v})$ that updates ρ at $\mathbf{v}$

$$\mathbf{Lift}(\rho, \mathbf{v})(\mathbf{u}) =$$

$$\begin{cases} \rho(\mathbf{u}), & \text{if } \mathbf{u} \neq \mathbf{v} \\ \max\{\rho(\mathbf{v}), \min_{\mathbf{v} \longrightarrow \mathbf{w}} \mathbf{Dom}(\rho, \mathbf{v}, \mathbf{w})\}, & \text{if } \mathbf{u} = \mathbf{v} \in \mathbf{V_0} \\ \max\{\rho(\mathbf{v}), \max_{\mathbf{v} \longrightarrow \mathbf{w}} \mathbf{Dom}(\rho, \mathbf{v}, \mathbf{w})\}, & \text{if } \mathbf{u} = \mathbf{v} \in \mathbf{V_1} \end{cases}$$

where $\mathbf{Dom}(\rho, \mathbf{v}, \mathbf{w})$ is the smallest value $\mathbf{m} \in \mathbf{M}_\top$ such that

Computing game progress measures

- Define an operator $\mathbf{Lift}(\rho, \mathbf{v})$ that updates ρ at $\mathbf{v}$

$$\mathbf{Lift}(\rho, \mathbf{v})(\mathbf{u}) =$$

$$\begin{array}{ll} \rho(\mathbf{u}), & \text{if } \mathbf{u} \neq \mathbf{v} \\ \max\{\rho(\mathbf{v}), \min_{\mathbf{v} \longrightarrow \mathbf{w}} \mathbf{Dom}(\rho, \mathbf{v}, \mathbf{w})\}, & \text{if } \mathbf{u} = \mathbf{v} \in \mathbf{V_0} \\ \max\{\rho(\mathbf{v}), \max_{\mathbf{v} \longrightarrow \mathbf{w}} \mathbf{Dom}(\rho, \mathbf{v}, \mathbf{w})\}, & \text{if } \mathbf{u} = \mathbf{v} \in \mathbf{V_1} \end{array}$$

where $\mathbf{Dom}(\rho, \mathbf{v}, \mathbf{w})$ is the smallest value $\mathbf{m} \in \mathbf{M}_\top$ such that

  - $\mathbf{m} \geq_{\mathbf{c(v)}} \rho(\mathbf{w})$, if $\mathbf{v} \in \mathbf{V_0}$

Computing game progress measures

- Define an operator $\mathbf{Lift}(\rho, \mathbf{v})$ that updates ρ at $\mathbf{v}$

$$\mathbf{Lift}(\rho, \mathbf{v})(\mathbf{u}) = \begin{cases} \rho(\mathbf{u}), & \text{if } \mathbf{u} \neq \mathbf{v} \\ \max\{\rho(\mathbf{v}), \min_{\mathbf{v} \longrightarrow \mathbf{w}} \mathbf{Dom}(\rho, \mathbf{v}, \mathbf{w})\}, & \text{if } \mathbf{u} = \mathbf{v} \in \mathbf{V_0} \\ \max\{\rho(\mathbf{v}), \max_{\mathbf{v} \longrightarrow \mathbf{w}} \mathbf{Dom}(\rho, \mathbf{v}, \mathbf{w})\}, & \text{if } \mathbf{u} = \mathbf{v} \in \mathbf{V_1} \end{cases}$$

where $\mathbf{Dom}(\rho, \mathbf{v}, \mathbf{w})$ is the smallest value $\mathbf{m} \in \mathbf{M}_\top$ such that

- $\mathbf{m} \geq_{\mathbf{c(v)}} \rho(\mathbf{w})$, if $\mathbf{v} \in \mathbf{V_0}$
- $\mathbf{m} >_{\mathbf{c(v)}} \rho(\mathbf{w})$ or $\mathbf{m} = \rho(\mathbf{w}) = \top$, if $\mathbf{v} \in \mathbf{V_1}$

Computing game progress measures

- Define an operator $\mathbf{Lift}(\rho, \mathbf{v})$ that updates ρ at $\mathbf{v}$

  $$\mathbf{Lift}(\rho, \mathbf{v})(\mathbf{u}) = \begin{cases} \rho(\mathbf{u}), & \text{if } \mathbf{u} \neq \mathbf{v} \\ \max\{\rho(\mathbf{v}), \min_{\mathbf{v} \to \mathbf{w}} \mathbf{Dom}(\rho, \mathbf{v}, \mathbf{w})\}, & \text{if } \mathbf{u} = \mathbf{v} \in \mathbf{V}_0 \\ \max\{\rho(\mathbf{v}), \max_{\mathbf{v} \to \mathbf{w}} \mathbf{Dom}(\rho, \mathbf{v}, \mathbf{w})\}, & \text{if } \mathbf{u} = \mathbf{v} \in \mathbf{V}_1 \end{cases}$$

  where $\mathbf{Dom}(\rho, \mathbf{v}, \mathbf{w})$ is the smallest value $\mathbf{m} \in \mathbf{M}_\top$ such that

  - $\mathbf{m} \geq_{\mathbf{c}(\mathbf{v})} \rho(\mathbf{w})$, if $\mathbf{v} \in \mathbf{V}_0$
  - $\mathbf{m} >_{\mathbf{c}(\mathbf{v})} \rho(\mathbf{w})$ or $\mathbf{m} = \rho(\mathbf{w}) = \top$, if $\mathbf{v} \in \mathbf{V}_1$

- **Lift** tries to raise the measure of each position in $\mathbf{V}_0$ above at least one neighbour and each position in $\mathbf{V}_1$ strictly above all neighbours

Computing game progress measures

- **Lift**$(\rho, \mathbf{v})$ is monotone for each $\mathbf{v}$

Computing game progress measures

- $\mathbf{Lift}(\rho, \mathbf{v})$ is monotone for each $\mathbf{v}$
- $\rho : \mathbf{V} \longrightarrow \mathbf{M}_{\top}$ is a game progress measure iff $\mathbf{Lift}(\rho, \mathbf{v}) \sqsubseteq \rho$ for each $\mathbf{v}$

Computing game progress measures

- $\mathbf{Lift}(\rho, \mathbf{v})$ is monotone for each $\mathbf{v}$
- $\rho : \mathbf{V} \longrightarrow \mathbf{M}_\top$ is a game progress measure iff $\mathbf{Lift}(\rho, \mathbf{v}) \sqsubseteq \rho$ for each $\mathbf{v}$
- Can compute simultaneous fixed point of all $\mathbf{Lift}(\rho, \mathbf{v})$ iteratively

Computing game progress measures

- $\mathbf{Lift}(\rho, \mathbf{v})$ is monotone for each $\mathbf{v}$
- $\rho : \mathbf{V} \longrightarrow \mathbf{M}_{\top}$ is a game progress measure iff $\mathbf{Lift}(\rho, \mathbf{v}) \sqsubseteq \rho$ for each $\mathbf{v}$
- Can compute simultaneous fixed point of all $\mathbf{Lift}(\rho, \mathbf{v})$ iteratively
  - Initialize $\mathbf{Lift}(\rho, \mathbf{v}) = (0, \ldots, 0)$ for all $\mathbf{v}$

Computing game progress measures

- $\mathbf{Lift}(\rho, \mathbf{v})$ is monotone for each $\mathbf{v}$
- $\rho : \mathbf{V} \longrightarrow \mathbf{M}_{\top}$ is a game progress measure iff $\mathbf{Lift}(\rho, \mathbf{v}) \sqsubseteq \rho$ for each $\mathbf{v}$
- Can compute simultaneous fixed point of all $\mathbf{Lift}(\rho, \mathbf{v})$ iteratively
  - Initialize $\mathbf{Lift}(\rho, \mathbf{v}) = (0, \ldots, 0)$ for all $\mathbf{v}$
  - So long as $\rho \sqsubset \mathbf{Lift}(\rho, \mathbf{v})$ for some $\mathbf{v}$, set $\rho = \mathbf{Lift}(\rho, \mathbf{v})$

Computing game progress measures

- $\mathbf{Lift}(\rho, \mathbf{v})$ is monotone for each $\mathbf{v}$
- $\rho : \mathbf{V} \longrightarrow \mathbf{M}_\top$ is a game progress measure iff $\mathbf{Lift}(\rho, \mathbf{v}) \sqsubseteq \rho$ for each $\mathbf{v}$
- Can compute simultaneous fixed point of all $\mathbf{Lift}(\rho, \mathbf{v})$ iteratively
  - Initialize $\mathbf{Lift}(\rho, \mathbf{v}) = (0, \ldots, 0)$ for all $\mathbf{v}$
  - So long as $\rho \sqsubset \mathbf{Lift}(\rho, \mathbf{v})$ for some $\mathbf{v}$, set $\rho = \mathbf{Lift}(\rho, \mathbf{v})$
- Computation takes space $\mathbf{O(dn \log n)}$

  To describe ρ, for each of $\mathbf{n}$ positions, store an element of $\mathbf{M}_\top$—$\mathbf{d}$ numbers in the range $\{0, \ldots, \mathbf{n}\}$, hence $\mathbf{n} \cdot \mathbf{d} \cdot \log \mathbf{n}$

Computing game progress measures

- $\mathbf{Lift}(\rho, \mathbf{v})$ is monotone for each $\mathbf{v}$
- $\rho : \mathbf{V} \longrightarrow \mathbf{M}_\top$ is a game progress measure iff $\mathbf{Lift}(\rho, \mathbf{v}) \sqsubseteq \rho$ for each $\mathbf{v}$
- Can compute simultaneous fixed point of all $\mathbf{Lift}(\rho, \mathbf{v})$ iteratively
  - Initialize $\mathbf{Lift}(\rho, \mathbf{v}) = (0, \ldots, 0)$ for all $\mathbf{v}$
  - So long as $\rho \sqsubset \mathbf{Lift}(\rho, \mathbf{v})$ for some $\mathbf{v}$, set $\rho = \mathbf{Lift}(\rho, \mathbf{v})$
- Computation takes space $\mathbf{O}(\mathbf{dn} \log \mathbf{n})$

  To describe ρ, for each of $\mathbf{n}$ positions, store an element of $\mathbf{M}_\top$—$\mathbf{d}$ numbers in the range $\{0, \ldots, \mathbf{n}\}$, hence $\mathbf{n} \cdot \mathbf{d} \cdot \log \mathbf{n}$
- Computation takes time $\mathbf{O}(\mathbf{d} \cdot \mathbf{m} \cdot \left(\frac{\mathbf{n}}{\lfloor \mathbf{d}/2 \rfloor}\right)^{\lfloor \mathbf{d}/2 \rfloor})$

  Analysis is a bit complicated

Strategy Improvement

- Given a pair of memoryless strategies (f_0, f_1) for players 0 and 1, associate a valuation to each position in the game

Strategy Improvement

- Given a pair of memoryless strategies $(\mathbf{f_0}, \mathbf{f_1})$ for players 0 and 1, associate a valuation to each position in the game
- Define an ordering on valuations and a notion of optimality

Strategy Improvement

- Given a pair of memoryless strategies $(\mathbf{f_0}, \mathbf{f_1})$ for players 0 and 1, associate a valuation to each position in the game
- Define an ordering on valuations and a notion of optimality
- Optimal valuations correspond to winning strategies

Strategy Improvement

- Given a pair of memoryless strategies $(\mathbf{f_0}, \mathbf{f_1})$ for players 0 and 1, associate a valuation to each position in the game
- Define an ordering on valuations and a notion of optimality
- Optimal valuations correspond to winning strategies
- If a valuation is not optimal for either player, improve it to get a better strategy

Strategy Improvement

- Given a pair of memoryless strategies (f_0, f_1) for players 0 and 1, associate a valuation to each position in the game
- Define an ordering on valuations and a notion of optimality
- Optimal valuations correspond to winning strategies
- If a valuation is not optimal for either player, improve it to get a better strategy
- Iteratively converge to an optimal (winning) strategy

Strategy Improvement

- Given a pair of memoryless strategies (f_0, f_1) for players 0 and 1, associate a valuation to each position in the game
- Define an ordering on valuations and a notion of optimality
- Optimal valuations correspond to winning strategies
- If a valuation is not optimal for either player, improve it to get a better strategy
- Iteratively converge to an optimal (winning) strategy
- Assumptions

Strategy Improvement

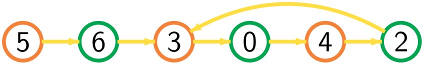
- Given a pair of memoryless strategies $(\mathbf{f_0}, \mathbf{f_1})$ for players 0 and 1, associate a valuation to each position in the game
- Define an ordering on valuations and a notion of optimality
- Optimal valuations correspond to winning strategies
- If a valuation is not optimal for either player, improve it to get a better strategy
- Iteratively converge to an optimal (winning) strategy
- Assumptions
  - Max-parity game—player 0 wins if largest infinitely occurring colour is even

Strategy Improvement

- Given a pair of memoryless strategies (f_0, f_1) for players 0 and 1, associate a valuation to each position in the game
- Define an ordering on valuations and a notion of optimality
- Optimal valuations correspond to winning strategies
- If a valuation is not optimal for either player, improve it to get a better strategy
- Iteratively converge to an optimal (winning) strategy
- Assumptions
  - Max-parity game—player 0 wins if largest infinitely occurring colour is even
  - All positions have distinct colours—assume positions and colours are both numbered $\{0, 1, \ldots, d\}$ so that position i has colour i

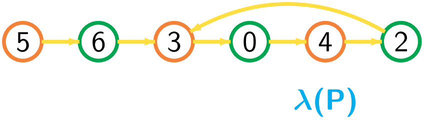
Valuations

A typical play **P** consistent with memoryless (f_0, f_1)



Valuations

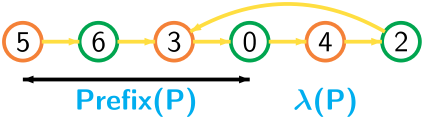
A typical play **P** consistent with memoryless $(\mathbf{f_0}, \mathbf{f_1})$



- $\lambda(\mathbf{P}) = \mathbf{4}$ — max colour in the loop

Valuations

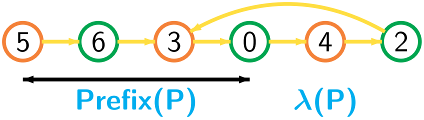
A typical play **P** consistent with memoryless (f_0, f_1)



- $\lambda(P) = 4$ — max colour in the loop
- $\pi(P) = \{5, 6\}$ — values higher than $\lambda(P)$ in $\text{Prefix}(P)$

Valuations

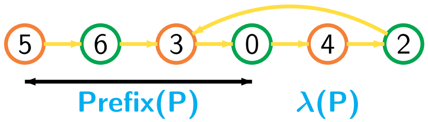
A typical play **P** consistent with memoryless (f_0, f_1)



- $\lambda(P) = 4$ — max colour in the loop
- $\pi(P) = \{5, 6\}$ — values higher than $\lambda(P)$ in **Prefix(P)**
- $\ell(P) = 4$ — length of **Prefix(P)**

Valuations

A typical play **P** consistent with memoryless (f_0, f_1)



- $\lambda(P) = 4$ — max colour in the loop
- $\pi(P) = \{5, 6\}$ — values higher than $\lambda(P)$ in **Prefix(P)**
- $\ell(P) = 4$ — length of **Prefix(P)**

Valuation

$\Theta : v \mapsto (\lambda(P), \pi(P), \ell(P))$ for some play **P** starting at **v**

Valuations . . .

- Strategy induced valuation

Valuations . . .

- Strategy induced valuation
  - Let (f_0, f_1) be memoryless strategies for players 0 and 1

Valuations . . .

- Strategy induced valuation
  - Let (f_0, f_1) be memoryless strategies for players 0 and 1
  - Assign $\Theta(v)$ according to path from v picked out by (f_0, f_1)

Valuations . . .

- Strategy induced valuation
  - Let (f_0, f_1) be memoryless strategies for players 0 and 1
  - Assign $\Theta(v)$ according to path from v picked out by (f_0, f_1)
- Locally progressive valuation

Valuations . . .

- Strategy induced valuation
  - Let (f_0, f_1) be memoryless strategies for players 0 and 1
  - Assign $\Theta(v)$ according to path from v picked out by (f_0, f_1)
- Locally progressive valuation
  - For each position u, there is a successor $u \longrightarrow v$ such that $\Theta(u)$ and $\Theta(v)$ refer to same path P

Valuations . . .

- Strategy induced valuation
  - Let (f_0, f_1) be memoryless strategies for players 0 and 1
  - Assign $\Theta(v)$ according to path from v picked out by (f_0, f_1)
- Locally progressive valuation
  - For each position u, there is a successor $u \longrightarrow v$ such that $\Theta(u)$ and $\Theta(v)$ refer to same path P
  - Write this as $u \rightsquigarrow v$

Valuations ...

- Strategy induced valuation
  - Let (f_0, f_1) be memoryless strategies for players 0 and 1
  - Assign $\Theta(v)$ according to path from v picked out by (f_0, f_1)
- Locally progressive valuation
  - For each position u, there is a successor $u \longrightarrow v$ such that $\Theta(u)$ and $\Theta(v)$ refer to same path P
  - Write this as $u \leadsto v$

Claim For any locally progressive valuation Θ, there are strategies (f_0, f_1) that induce Θ

Valuations . . .

- Strategy induced valuation
  - Let (f_0, f_1) be memoryless strategies for players 0 and 1
  - Assign $\Theta(v)$ according to path from v picked out by (f_0, f_1)
- Locally progressive valuation
  - For each position u, there is a successor $u \longrightarrow v$ such that $\Theta(u)$ and $\Theta(v)$ refer to same path P
  - Write this as $u \leadsto v$

Claim For any locally progressive valuation Θ, there are strategies (f_0, f_1) that induce Θ

- From any locally progressive valuation Θ, we can extract a pair of strategies (f_0, f_1) that induce Θ

Valuations . . .

- Strategy induced valuation
  - Let (f_0, f_1) be memoryless strategies for players 0 and 1
  - Assign $\Theta(v)$ according to path from v picked out by (f_0, f_1)
- Locally progressive valuation
  - For each position u, there is a successor $u \longrightarrow v$ such that $\Theta(u)$ and $\Theta(v)$ refer to same path P
  - Write this as $u \rightsquigarrow v$

Claim For any locally progressive valuation Θ, there are strategies (f_0, f_1) that induce Θ

- From any locally progressive valuation Θ, we can extract a pair of strategies (f_0, f_1) that induce Θ
- From any pair of strategies (f_0, f_1) we can derive a locally progressive valuation Θ

Ordering valuations

- Order $\Theta(u) = (w, P, \ell)$ and $\Theta(v) = (x, Q, m)$ lexicographically

Ordering valuations

- Order $\Theta(u) = (w, P, \ell)$ and $\Theta(v) = (x, Q, m)$ lexicographically
- Linear order on colours (positions) $\{0, 1, \ldots, 2k\}$

  $(2k{-}1) \prec (2k{-}3) \prec \cdots \prec 3 \prec 1 \prec 0 \prec 2 \prec \cdots < 2k$

Ordering valuations

- Order $\Theta(u) = (w, P, \ell)$ and $\Theta(v) = (x, Q, m)$ lexicographically
- Linear order on colours (positions) $\{0, 1, \ldots, 2k\}$

  $(2k{-}1) \prec (2k{-}3) \prec \cdots \prec 3 \prec 1 \prec 0 \prec 2 \prec \cdots < 2k$
- Linear order on sets of colours P and Q

  $P \prec Q$ iff $\max(P \setminus Q) \prec \max(Q \setminus P)$

Ordering valuations

- Order $\Theta(u) = (w, P, \ell)$ and $\Theta(v) = (x, Q, m)$ lexicographically
- Linear order on colours (positions) $\{0, 1, \ldots, 2k\}$

  $(2k{-}1) \prec (2k{-}3) \prec \cdots \prec 3 \prec 1 \prec 0 \prec 2 \prec \cdots < 2k$
- Linear order on sets of colours P and Q

  $P \prec Q$ iff $\max(P \setminus Q) \prec \max(Q \setminus P)$
- Order on l and m is normal $\leq$

Optimal valuations

A valuation Θ is optimal if we have:

> Whenever $u \rightsquigarrow v$, among successors of u, $\Theta(v)$ is largest value with respect to $\prec$

Optimal valuations

A valuation Θ is optimal if we have:

> Whenever $u \rightsquigarrow v$, among successors of u, $\Theta(v)$ is largest value with respect to $\prec$

Lemma

If Θ is an optimal valuation for player 0 (player 1), the corresponding strategy is winning for player 0 (player 1).

Strategy improvement

- Begin with arbitrary memoryless strategies $(\mathbf{f_0}, \mathbf{f_1})$

Strategy improvement

- Begin with arbitrary memoryless strategies $(\mathbf{f_0}, \mathbf{f_1})$
- Construct induced valuations

Strategy improvement

- Begin with arbitrary memoryless strategies (f_0, f_1)
- Construct induced valuations
- If the strategy is not optimal for player 0 (player 1), pick a nonoptimal position and improve it

Strategy improvement

- Begin with arbitrary memoryless strategies $(\mathbf{f_0}, \mathbf{f_1})$
- Construct induced valuations
- If the strategy is not optimal for player 0 (player 1), pick a nonoptimal position and improve it
- Repeat until both players have an optimal strategy

Strategy improvement

- Begin with arbitrary memoryless strategies (f_0, f_1)
- Construct induced valuations
- If the strategy is not optimal for player 0 (player 1), pick a nonoptimal position and improve it
- Repeat until both players have an optimal strategy

Claim This procedure converges.

Strategy improvement

- Begin with arbitrary memoryless strategies (f_0, f_1)
- Construct induced valuations
- If the strategy is not optimal for player 0 (player 1), pick a nonoptimal position and improve it
- Repeat until both players have an optimal strategy

  Claim This procedure converges.
- No theoretical bound is known on the complexity of convergence.

References

- Marcin Jurdiński
  Small Progress Measures for Solving Parity Games
  Proc STACS 2000
  Springer LNCS 1770 (2000) 290–301
- Marcin Jurdiński and Jens Vöge
  A Discrete Strategy Improvement Algorithm for Solving Parity Games
  Proc CAV 2000
  Springer LNCS 1855 (2000) 202–215
- Hartmut Klauck
  Algorithms for Parity Games
  in Erich Grädel, Wolfgang Thomas, Thomas Wilke (Eds.):
  Automata, Logics, and Infinite Games: A Guide to Current Research,
  Springer LNCS 2500 (2002) 107–129