A Discrete Strategy Improvement Algorithm for Solving Parity Games

Jens Vöge
Lehrstuhl für Informatik VII
RWTH Aachen
Germany

Marcin Jurdziński
BRICS
University of Aarhus
Denmark

Chicago, USA, 19 July 2000

Complexity of parity games — motivations

- Equivalent to modal μ-calculus model checking
  [Emerson, Jutla, Sistla 1993; Stirling 1995]

Model checking: does $\mathcal{K} \models \varphi$ hold?	$\longmapsto$	Solving a parity game: who is the winner in $\mathcal{G}_{\mathcal{K},\varphi}$?

reduction in time $O\big(|\mathcal{K}| \cdot |\varphi|\big)$

- Intriguing complexity-theoretic status
  - in **NP** $\cap$ **co-NP** [EJS'93] (even in **UP** $\cap$ **co-UP** [J'98])
  - no polynomial time algorithm known
    [EL'86, . . . , EJS'93, BCJLM'94, Sei'96, J'00]
  - parity games $\leq_m^{\text{log-space}}$ mean payoff games $\leq_m^{\text{log-space}}$ discounted payoff games $\leq_m^{\text{log-space}}$ simple stochastic games
    [Condon'92, Puri'95, ZP'96]

Strategy improvement algorithms — history

1966 Hoffman-Karp's algorithm for stochastic games
received a lot of attention in Operations Research community

1995 Puri's algorithm for discounted payoff games

Drawbacks of Puri's algorithm:

- Inefficient in practice
  - solving linear programming instances
  - high precision arithmetic
- Hard to analyze/understand
  - manipulates real number encodings of discrete values
  - proof of correctness uses continuous methods (e.g., Banach fixed point theorem)

Discrete strategy improvement algorithm

We alleviate drawbacks of Puri's algorithm

- Fast implementation
  - $O(n \cdot m)$ discrete algorithm for strategy improvement step
- Hope for easier analysis/better understanding:
  - manipulates discrete values explicitly
  - proof of correctness uses only discrete arguments

Experimental evidence: small number of strategy improvement steps

Open problem: is it a polynomial time algorithm?

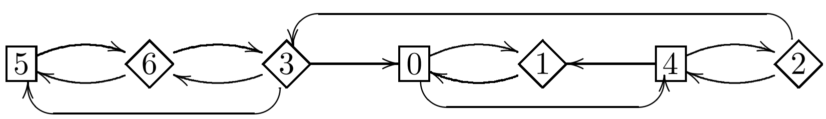
Plan

1. Definition of parity games
2. Solving parity games
   (a) as a decision problem
   (b) as an optimization problem
3. Strategy Improvement Algorithm
   (a) generic idea
   (b) our implementation
4. Time complexity
5. Open problems

Parity games — definition

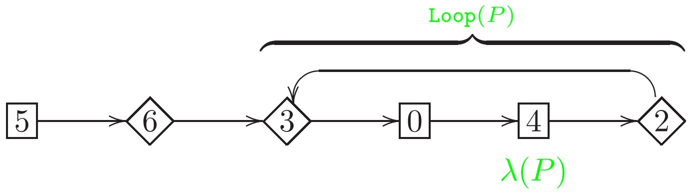
$$G = \big(V, E, (M_{\mathtt{Even}}, M_{\mathtt{Odd}})\big)$$

- $V = \{0, 1, 2, \ldots, n\} = M_{\mathtt{Even}} \uplus M_{\mathtt{Odd}}$
- $\square \in M_{\mathtt{Even}}$; $\diamond \in M_{\mathtt{Odd}}$



Winning play

Play P:



$\texttt{Loop}(P) = \{0, 2, 3, 4\}$ $\qquad$ $\lambda(P) = \max\big(\{0, 2, 3, 4\}\big) = 4$

Loop value $\lambda(P)$ of a play P is defined by

$$\lambda(P) \;=\; \max\big(\texttt{Loop}(P)\big)$$

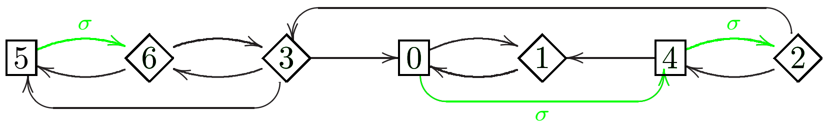
Play P is a winning play for **Even** iff

$$\lambda(P) \text{ is even}$$

Strategies

Function $\sigma : M_{\texttt{Even}} \to V$ is a strategy for **Even** iff

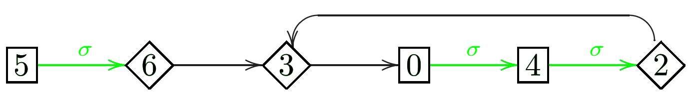
$$\big(v, \sigma(v)\big) \in E \text{ for every } v \in M_{\texttt{Even}}$$



Play $P = \langle v_1, v_2, \ldots, v_k \rangle$ is consistent with strategy σ iff

$$v_{i+1} = \sigma(v_i) \text{ for every } v_i \in M_{\texttt{Even}}$$

P:



Winning strategies

$\texttt{Plays}_\sigma(v)$ is defined to be the set of plays

– starting from v, and

– consistent with σ

Strategy σ is a winning strategy for $\texttt{Even}$ from v iff
every play $P \in \texttt{Plays}_\sigma(v)$ is winning for $\texttt{Even}$

Solving parity games — decision problem

The winning set

$$W_{\texttt{Even}} = \{\, v \in V \;:\; \text{there is a winning strategy for } \texttt{Even} \text{ from } v \,\}$$

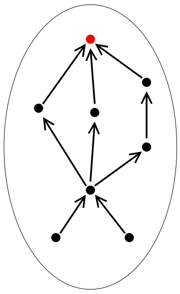
The problem of solving parity games

Given: a parity game $G = \big(V, E, (M_{\texttt{Even}}, M_{\texttt{Odd}})\big)$

Find: the winning set $W_{\texttt{Even}} \subseteq V$

Solving parity games — optimization problem

(Strategies, $\sqsubseteq$)



$\sqsubseteq$ is a partial order on `Strategies`

Postulates for (Strategies, $\sqsubseteq$):

1. The $\sqsubseteq$-maximum strategy exists
2. The $\sqsubseteq$-maximum element is a strategy winning from every vertex in $W_{\texttt{Even}}$

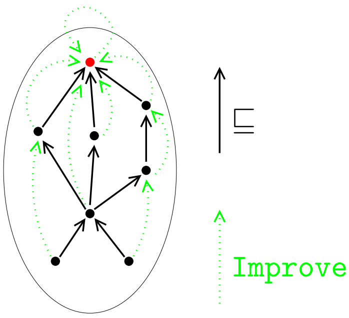
The problem of solving parity games

Given: a parity game $G = \big(V, E, (M_{\texttt{Even}}, M_{\texttt{Odd}})\big)$

Find: the $\sqsubseteq$-maximum strategy

Generic Strategy Improvement Algorithm

(Strategies, $\sqsubseteq$)



Postulates for a function

`Improve` : `Strategies` $\to$ `Strategies`

1. (Strategy Improvement)
   If σ is not the $\sqsubseteq$-maximum strategy
   then $\sigma \sqsubseteq$ `Improve`(σ)
2. (Optimum Strategy)
   If σ is the $\sqsubseteq$-maximum strategy
   then `Improve`$(\sigma) = \sigma$

Strategy Improvement Algorithm:

```
pick a strategy σ for player Even
while σ ≠ Improve(σ)
    do σ := Improve(σ)
```

Ingredients of Strategy Improvement Algorithm

In this talk:

1. Definition of a partial order $\sqsubseteq$ on `Strategies`
2. Definition of a function `Improve` : `Strategies` $\to$ `Strategies`

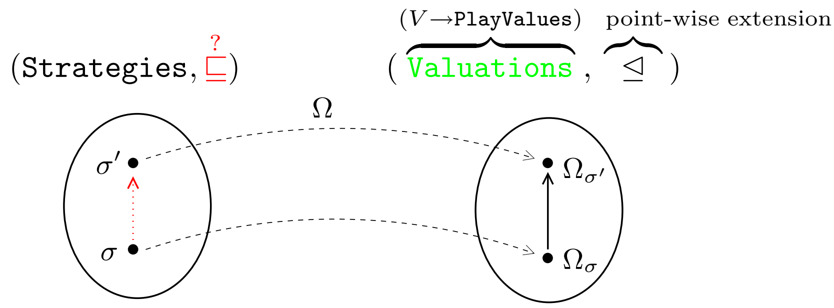
In (full versions of) the paper:

1. Proofs of postulates for (`Strategies`, $\sqsubseteq$)
2. Proofs of postulates for `Improve`:
   (a) Proof of Strategy Improvement Lemma
   (b) Proof of Optimum Strategy Lemma
3. Efficient implementation of `Improve`

Our proposal for $\sqsubseteq$-order on `Strategies`

Assume we are given:

1. (`PlayValues`, $\trianglelefteq$): a linear order $\trianglelefteq$ on `PlayValues`
2. Θ : `Plays` $\to$ `PlayValues`: value of a play



$\Omega_\sigma : V \to$ `PlayValues`

$$\Omega_\sigma(v) = \min\nolimits^{\trianglelefteq}\{\ \Theta(P)\ :\ P \in \texttt{Plays}_\sigma(v)\ \}$$

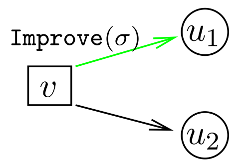
Intuition: Ω_σ is the value of the best counter-strategy against σ

Our proposal for Improve function on Strategies

$$\texttt{Improve} : \texttt{Strategies} \rightarrow \texttt{Strategies}$$

$$\texttt{Improve}(\sigma) : M_{\texttt{Even}} \rightarrow V$$

$$\big[\texttt{Improve}(\sigma)\big](v) \;=\; \text{the successor of } v \text{ maximizing } \Omega_\sigma \text{ (w.r.t. } \trianglelefteq)$$



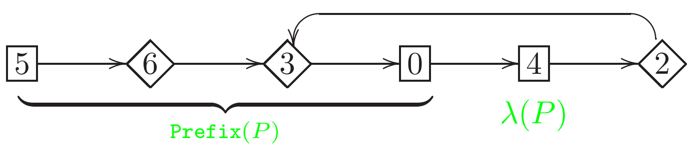
because $\Omega_\sigma(u_1) \triangleright \Omega_\sigma(u_2)$

Improvement step for strategy σ in a nutshell:

1. global minimization: find Ω_σ, the best counter-strategy against σ
2. local maximization: point $\texttt{Improve}(\sigma)$ to $\trianglelefteq$-maximum Ω_σ values

PlayValues and function $\Theta : \texttt{Plays} \rightarrow \texttt{PlayValues}$

Play P:



$\texttt{Prefix}(P) = \{0, 3, 5, 6\}$

$\lambda(P) = 4$, $\pi(P) = \{5, 6\}$, $\#(P) = 4$

Primary path value	$\pi(P)$	$=$	$\texttt{Prefix}(P) \cap \{\, v : v > \lambda(P) \,\}$
Secondary path value	$\#(P)$	$=$	$\big\lvert\texttt{Prefix}(P)\big\rvert$

Value of a play function $\Theta : \texttt{Plays} \rightarrow \overbrace{\texttt{PlayValues}}^{V \times \wp(V) \times \mathbb{N}}$ is defined by:

$$\Theta(P) = \big(\lambda(P), \pi(P), \#(P)\big)$$

Our proposal for $\trianglelefteq$ order on `PlayValues`

$$\texttt{PlayValues} \subseteq V \times \wp(V) \times \mathbb{N}$$

$\trianglelefteq$ on `PlayValues` is the lexicographic order on $V \times \wp(V) \times \mathbb{N}$

- We define a $\preceq$ linear order on loop values V
- We define a $\preceq$ linear order on primary path values $\wp(V)$
- We use standard $\leq$ linear order on secondary path values $\mathbb{N}$

The $\preceq$ linear orders on V and $\wp(V)$

Definition (The $\preceq$ linear order on loop values V)

$$(2k-1) \prec \cdots \prec 5 \prec 3 \prec 1 \prec 0 \prec 2 \prec 4 \prec 6 \prec \cdots \prec 2k$$

Definition (The $\preceq$ linear order on primary path values $\wp(V)$)

$$P \preceq Q \quad \text{iff} \quad \texttt{FirstDiff}(P;Q) \preceq \texttt{FirstDiff}(Q;P)$$

Example

$$\begin{aligned} P &= \{\ 7 > 6 > 5 > 4\ \} \\ Q &= \{\ 7 > 6 > 4\ \} \\ R &= \{\ 7 > 6 > 4 > 2\ \} \end{aligned}$$

Proof techniques

1. Efficient implementation of `Improve`; computing Ω_σ
   - solving one-player games
     (for player `Odd`: minimization instead of maximization)
   - solving special instances of shortest paths problem
2. Strategy Improvement and Optimum Strategy Lemmas
   - local characterization of Ω_σ (progressive valuation)
   - relaxations of valuations (under- and over-valuations)
     Lemma 1. If Ξ is an under-valuation for G_σ then $\Xi \trianglelefteq \Omega_\sigma$
     Lemma 2. If Ξ is an over-valuation for G_τ then $\mho_\tau \trianglelefteq \Xi$
   - application of lemmas 1. and 2.
     Prop.1. Ω_σ is an under-valuation for $G_{\texttt{Improve}(\sigma)}$
     Prop.2. If $\texttt{Improve}(\sigma) = \sigma$ then Ω_σ is an over-valuation for $G_{\overline{\sigma}}$

Time complexity

Parameters of interest

- The time needed to perform a single strategy improvement step
  - A discrete $O(n \cdot m)$ time algorithm
    (efficient implementation in a companion paper [SV'00])
- The number of strategy improvement steps
  - An obvious $\prod_{v \in M_{\text{Even}}}$ out-deg(v) upper bound
  - Prop. $O(n^3)$ strategy improvement steps suffice for one-player parity games (cf. [Melekopoglou, Condon 1994])
  - Prop. There exists a policy of improvement at one vertex at a time terminating in at most n steps (cf. [J'00])
  - Prop. There are only $O(n^2)$ substantial improvement steps
    Experimental evidence. Small, often $O(1)$ number of non-substantial improvement steps. (see [SV'00])

Open problems

1. Does our algorithm with the standard improvement policy terminate in polynomial number of strategy improvement steps?

   If not: exhibit families of hard examples

2. Are there polynomial time improvement policies for which our algorithm terminates in polynomial number of strategy improvement steps?

   If not: exhibit families of hard examples

3. Define and study other partial orders $\sqsubseteq$ on `Strategies` and other `Improve` operators

4. Develop other algorithms than strategy improvement algorithm for the optimization problem we have defined