Algorithms for Parity Games

Piotr Danilewski

May 15, 2008

Practical use of Parity Games

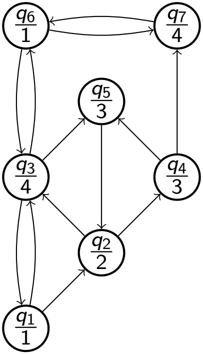
- Modal μ-calculus model checking
- Synthesis and satisfiability checking for reactive systems
- Module checking

What are Parity Games?

Parity Graph

$$G = (V, E, p)$$

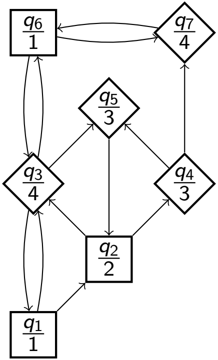
$$p : V \longrightarrow \mathbb{N}$$



Parity Game

Nodes assigned to players $\boxed{\text{A}}$ and $\diamond$B .

$$V = V_A \cup V_B$$

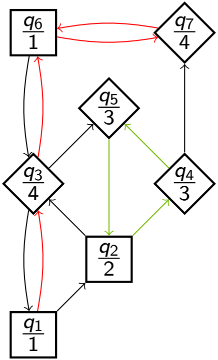


Play

Play - infinite path $\pi = (v_0, v_1, v_2, ...)$

$$\pi_1 = (q_1, q_3, q_6, q_7, q_6, q_7, ...)$$

$$\pi_2 = (q_2, q_4, q_5, q_2, q_4, q_5, ...)$$



Winning condition

Let P denote minimal priority which repeats itself infinitely often.

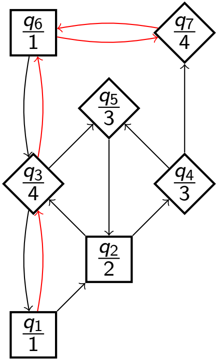
- If P is odd then player $\boxed{A}$ wins.
- If P is even then player $\diamond$B wins.

Winning condition

$$\pi_1 = (q_1, q_3, q_6, q_7, q_6, q_7, \ldots)$$

$$(1, 4, 1, 4, 1, 4, 1, 4\ldots)$$

$$P = 1 \Rightarrow \boxed{\text{A}} \text{ wins.}$$

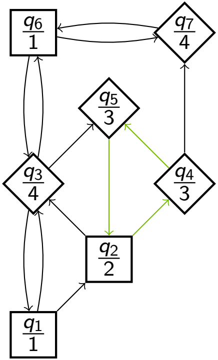


Winning condition

$$\pi_2 = (q_2, q_4, q_5, q_2, q_4, q_5, \dots)$$

$$(2, 3, 3, 2, 3, 3, 2, 3\dots)$$

$$P = 2 \Rightarrow \Diamond\!\!\text{B} \text{ wins.}$$

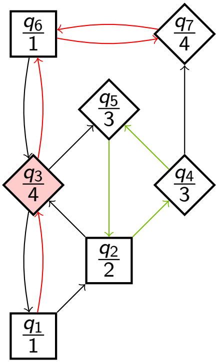


When can we win?

Ideal playing

We assume players do not make mistakes.

- π_1 is invalid under this assumption.
  At vertex q_3 player B should have chosen q_5.
- π_2 is valid. Choosing q_3 at q_2 would not help player A.



Memoryless property

We do not have to know how we reached certain vertex in order to deduct how to play.

Memoryless strategy representation

Strategy does not change over time.

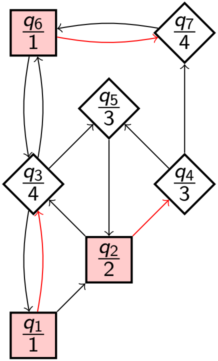
$$s_A : V_A \longrightarrow V$$

$$s_B : V_B \longrightarrow V$$

s_A, s_B point to successor picked by players A and B respectively.

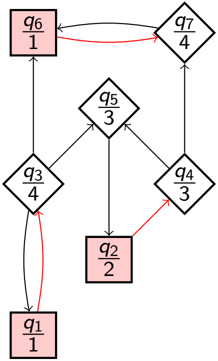
Memoryless strategy representation

$$s_A(v) := \begin{cases} q_3 \text{ if } v = q_1 \\ q_4 \text{ if } v = q_2 \\ q_7 \text{ if } v = q_6 \end{cases}$$



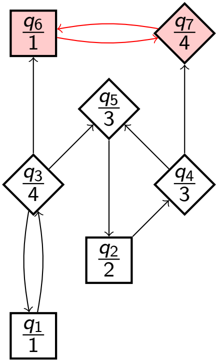
Strategy graph

G_A - Graph G were edges outgoing from V_A are limited to only those chosen by s_A.



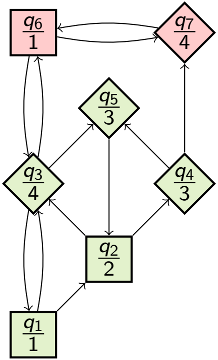
Winning condition in strategy graph

Player A wins if for given vertex v all reachabe cycles in G_A are odd.



Winning sets partition

Every parity game graph can be partitioned into winning sets W_A and W_B.



Algorithms

- Enrich graph with additional information and deduct the best strategy
- Choose some strategy and then improve it

Naive Algorithm

The naive algorithm

- Consider all possible strategies s_A.
- Consider all possible counter-strategies s_B.
- Pick the best s_A

Time complexity of the naive algorithm

$$O\left(\prod_{v \in V} \deg v\right)$$

In case of full graph:

$$O\left(|V|^{|V|}\right)$$

Jurdzinski's algorithm

Parity Progress Measure

Store additional tuple $\mathbb{N}^{c+1}$ at each vertex where c is maximal priority.
Comparison operators: $<_i, \leq_i, =_i, \geq_i, >_i$ - lexicographic operators on $i+1$ first elements.

$$(2, 3, 0, 0) >_1 (2, 2, 4, 1)$$

$$(2, 3, 0, 0) =_0 (2, 2, 4, 1)$$

$$(0, 1, 0, 0) <_1 (1, 0, 0, 0)$$

Parity Progress Measure

Function $\rho : V \longrightarrow \mathbb{N}^{c+1}$ is a *Parity Progress Measure* if:

$$\forall (v, w) \in E : \rho(v) \geq_{p(v)} \rho(w)$$

and if $p(v)$ is odd

$$\forall (v, w) \in E : \rho(v) >_{p(v)} \rho(w)$$

Parity Progress Measure

If Parity Progress Measure ρ exists then graph G must have only even cycles.
Otherwise, let i be minimal odd priority in some cycle. Then:

$$\rho(v_1) >_i \rho(v_2) \geq_i \rho(v_3) ... \geq_i \rho(v_1)$$

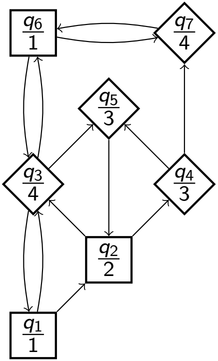
Parity Progress Measure

$$V_i := p^{-1}(i)$$

$$M_G := \mathbb{N}_0 \times \mathbb{N}_{|V_1|} \times \mathbb{N}_0 \times \mathbb{N}_{|V_3|} \times \mathbb{N}_0 \times \ldots \times \mathbb{N}_{|V_c|}$$

Codomain restriction

$$\begin{cases} V_0 := \emptyset \\ V_1 := \{q_1, q_6\} \\ V_2 := \{q_2\} \\ V_3 := \{q_4, q_5\} \\ V_4 := \{q_3, q_7\} \end{cases}$$



$$M_G = \mathbb{N}_0 \times \mathbb{N}_2 \times \mathbb{N}_0 \times \mathbb{N}_2 \times \mathbb{N}_0$$

Game Parity Progress Measure

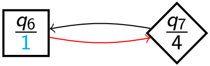
So far - we worked on graphs with even cycles only.
Now - we want to include all graphs and vertex assignment to players.
Add highest element T:

$$M_G^T = M_G \cup \{T\}$$

T means we cannot fit any other value because we reach an odd cycle

Game Parity Progress Measure

Let $\rho : V \longrightarrow M_G^T$ be *any* function.
Small progress function:
$Prog(\rho, v, w) :=$ least $m \in M_G^T : m \geq_{p(v)} \rho(w)$, and inequality must be strict if $p(v)$ is odd.

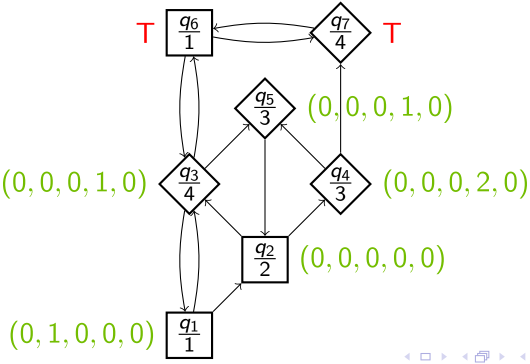
$(0, 1, 0, 0, 0)$  $(0, 1, 0, 1, 0)$

$Prog(\rho, q_6, q_7) = \{$ least $m >_1 (0, 1, 0, 1, 0)\} = (0, 2, 0, 0, 0)$

Game Parity Progress Measure

Game Parity Progress Measure is a function $\rho : V \longrightarrow M_G^T$ such that for all $v \in V$:

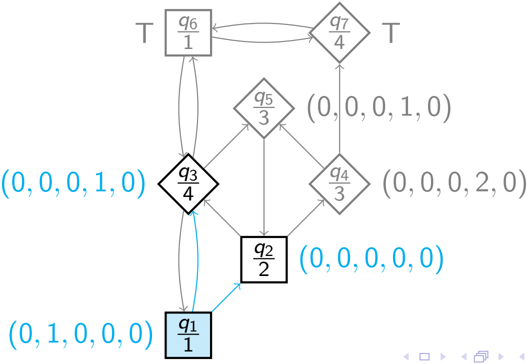
- $v \in V_A \Rightarrow \forall (v,w) \in E : \rho(v) \geq_{p(v)} Prog(\rho, v, w)$
- $v \in V_B \Rightarrow \exists (v,w) \in E : \rho(v) \geq_{p(v)} Prog(\rho, v, w)$



Game Parity Progress Measure

Game Parity Progress Measure is a function $\rho : V \longrightarrow M_G^T$ such that for all $v \in V$:

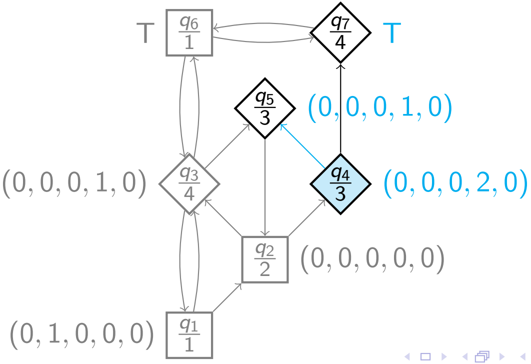
- $v \in V_A \Rightarrow \forall (v, w) \in E : \rho(v) \geq_{p(v)} Prog(\rho, v, w)$
- $v \in V_B \Rightarrow \exists (v, w) \in E : \rho(v) \geq_{p(v)} Prog(\rho, v, w)$



Game Parity Progress Measure

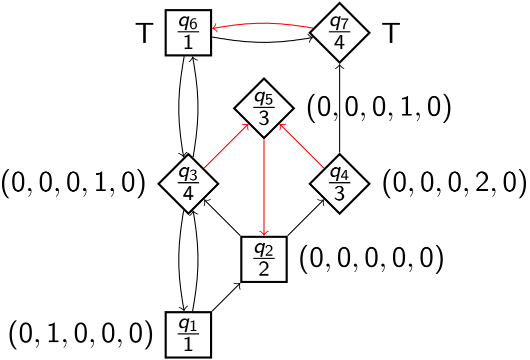
Game Parity Progress Measure is a function $\rho : V \longrightarrow M_G^T$ such that for all $v \in V$:

- $v \in V_A \Rightarrow \forall (v, w) \in E : \rho(v) \geq_{p(v)} Prog(\rho, v, w)$
- $v \in V_B \Rightarrow \exists (v, w) \in E : \rho(v) \geq_{p(v)} Prog(\rho, v, w)$



Strategy from Game Parity Progress Measure

Given ρ player B forms strategy s_B by minimalising its value.

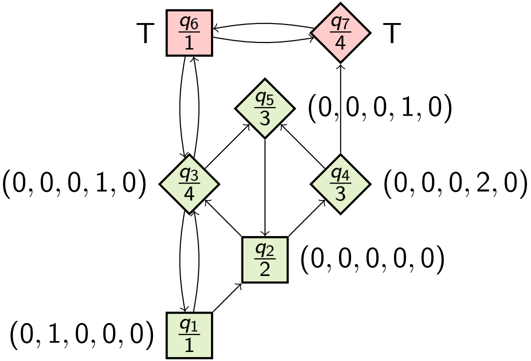


Winning sets from Game Parity Progress Measure

If ρ is a *minimal* Game Parity Progress Measure:

$$W_B = \{v \in V : \rho(v) \neq T\}$$

$$W_A = \{v \in V : \rho(v) = T\}$$



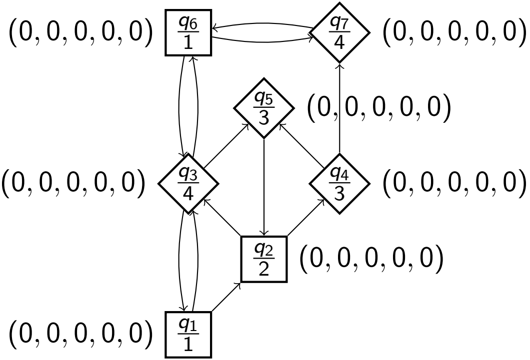
Computing the minimal Game Parity Progress Measure

- Start by assigning $\rho(v) := (0, 0, \ldots, 0)$ to all vertices
- Increment each vertex which violates the Game Parity Progress Measure constraints

$$Lift(\rho, v)(u) := \begin{cases} \rho(u) \Leftarrow u \neq v \\ \max\left(\rho(v), \min_{(v,w)\in E} Prog(\rho, v, w)\right) \Leftarrow u = v \in V_B \\ \max\left(\rho(v), \max_{(v,w)\in E} Prog(\rho, v, w)\right) \Leftarrow u = v \in V_A \end{cases}$$

Run of Jurdzinski's Algorithm

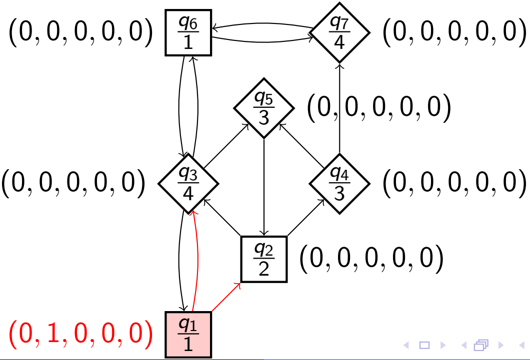
Run of Jurdzinski's algorithm (step 0)



Run of Jurdzinski's algorithm (step 1)

$$Lift(\rho, q_1)(q_1) = \max\left((0,0,0,0,0), \max_{(v,w)\in E} Prog(\rho, q_1, w)\right)$$

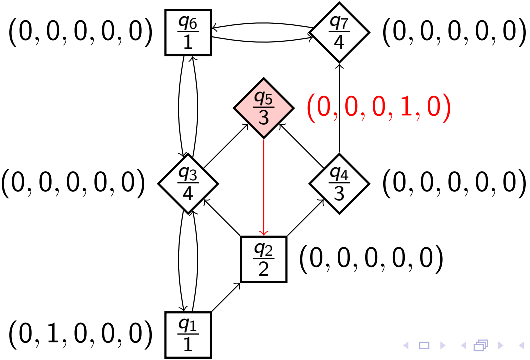
$$Prog(\rho, q_1, w) := \text{ least } m \in M_G^T : m >_1 \rho(w)$$



Run of Jurdzinski's algorithm (step 2)

$$Lift(\rho, q_5)(q_5) = \max\left((0,0,0,0,0)\,,\ \min_{(v,w)\in E} Prog(\rho, q_5, w)\right)$$

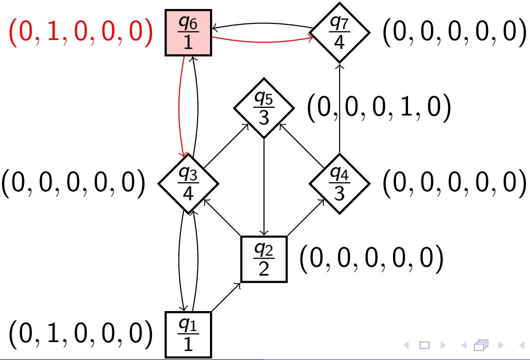
$$Prog(\rho, q_5, w) := \text{ least } m \in M_G^T : m >_3 \rho(w)$$



Run of Jurdzinski's algorithm (step 3)

$$Lift(\rho, q_6)(q_6) = \max\left((0,0,0,0,0), \max_{(v,w)\in E} Prog(\rho, q_6, w)\right)$$

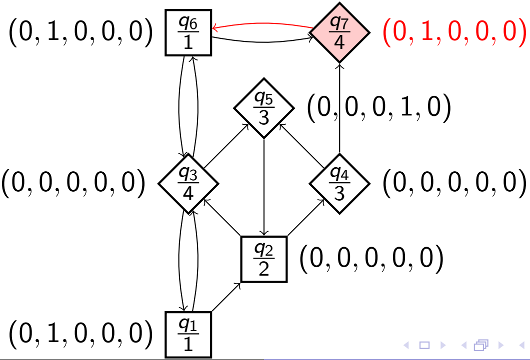
$$Prog(\rho, q_6, w) := \text{ least } m \in M_G^T : m >_1 \rho(w)$$



Run of Jurdzinski's algorithm (step 4)

$$Lift(\rho, q_7)(q_7) = \max\left((0,0,0,0,0)\,,\ \min_{(v,w)\in E} Prog(\rho, q_7, w)\right)$$

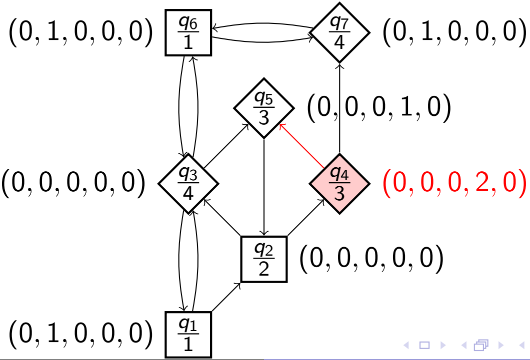
$$Prog(\rho, q_7, w) := \text{ least } m \in M_G^T : m \geq_4 \rho(w)$$



Run of Jurdzinski's algorithm (step 5)

$$Lift(\rho, q_4)(q_4) = \max\left((0,0,0,0,0),\ \min_{(v,w)\in E} Prog(\rho, q_4, w)\right)$$

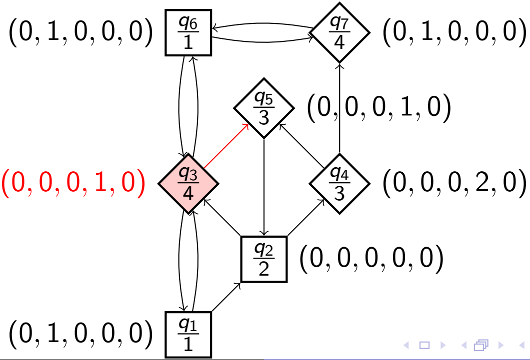
$$Prog(\rho, q_4, w) := \text{ least } m \in M_G^T : m >_3 \rho(w)$$



Run of Jurdzinski's algorithm (step 6)

$$Lift(\rho, q_3)(q_3) = \max\left((0,0,0,0,0), \min_{(v,w)\in E} Prog(\rho, q_3, w)\right)$$

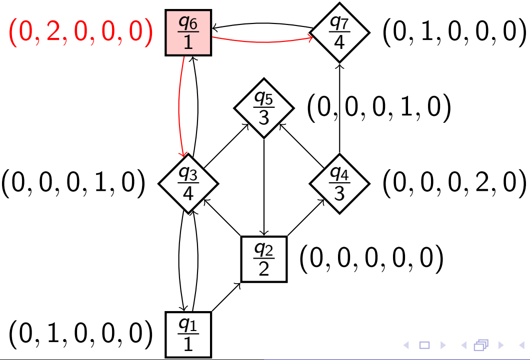
$$Prog(\rho, q_3, w) := \text{ least } m \in M_G^T : m \geq_4 \rho(w)$$



Run of Jurdzinski's algorithm (step 7)

$$Lift(\rho, q_6)(q_6) = \max\left((0,1,0,0,0)\,,\ \max_{(v,w)\in E} Prog(\rho, q_6, w)\right)$$

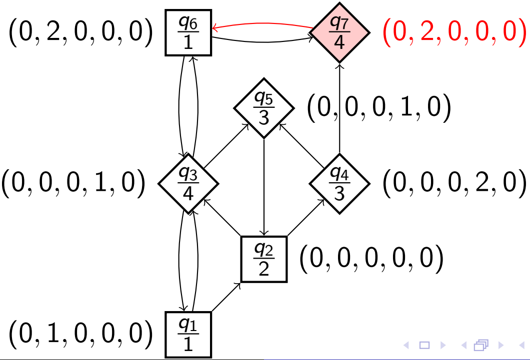
$$Prog(\rho, q_6, w) := \text{ least } m \in M_G^T : m >_1 \rho(w)$$



Run of Jurdzinski's algorithm (step 8)

$$Lift(\rho, q_7)(q_7) = \max\left((0,1,0,0,0), \min_{(v,w)\in E} Prog(\rho, q_7, w)\right)$$

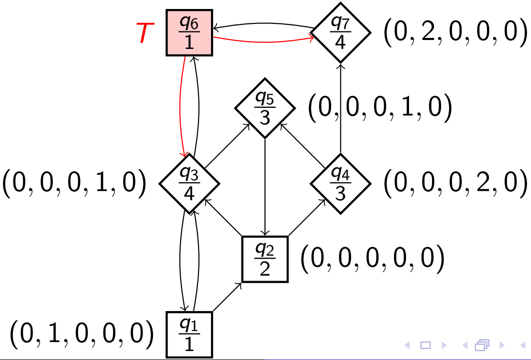
$$Prog(\rho, q_7, w) := \text{ least } m \in M_G^T : m \geq_4 \rho(w)$$



Run of Jurdzinski's algorithm (step 9)

$$Lift(\rho, q_6)(q_6) = \max\left((0,2,0,0,0), \max_{(v,w)\in E} Prog(\rho, q_6, w)\right)$$

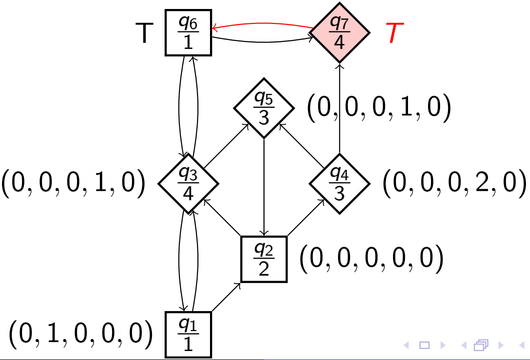
$$M_G^T = \mathbb{N}_0 \times \mathbb{N}_2 \times \mathbb{N}_0 \times \mathbb{N}_2 \times \mathbb{N}_0 \cup \{T\}$$



Run of Jurdzinski's algorithm (step 10)

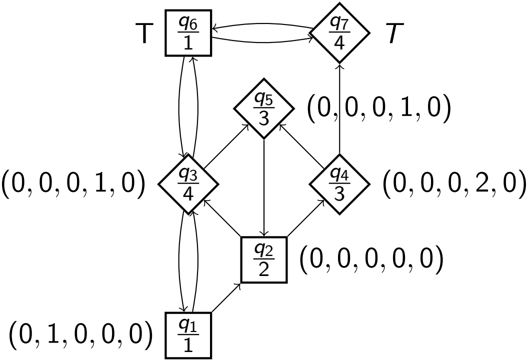
$$Lift(\rho, q_7)(q_7) = \max\left((0,2,0,0,0)\,,\ \min_{(v,w)\in E} Prog(\rho, q_7, w)\right)$$

$$M_G^T = \mathbb{N}_0 \times \mathbb{N}_2 \times \mathbb{N}_0 \times \mathbb{N}_2 \times \mathbb{N}_0 \cup \{T\}$$



Run of Jurdzinski's algorithm (step 11)

Observe Lift operation cannot perform any more changes.
End of run.



Space complexity of Jurdzinski's algorithm

$$O(c\,|V|)$$

Time complexity of Jurdzinski's algorithm

Time per single lift operation: $O(c \deg v)$
Total time:

$$O(\sum_{v \in V} |M_G| \, c \deg v) = O\left(c\,|E|\,|M_G|\right)$$

$$|M_G| = \prod_{i=1}^{\lceil \frac{c}{2} \rceil} (|V_{2i-1}| + 1) < \left(\frac{\sum_{i=1}^{\lceil \frac{c}{2} \rceil} (|V_{2i-1}| + 1)}{\lceil \frac{c}{2} \rceil} \right)^{\lceil \frac{c}{2} \rceil} \leq \left(\frac{|V|}{\lceil \frac{c}{2} \rceil} \right)^{\lceil \frac{c}{2} \rceil}$$

Finally

$$O\left(c\,|E| \left(\frac{|V|}{\lceil \frac{c}{2} \rceil} \right)^{\lceil \frac{c}{2} \rceil} \right)$$

There are other algorithms, for example:

- McNaughton's algorithm
- Sven's algorithm

We know that Parity Games problem

- is NP $\cap$ co-NP
- is UP $\cap$ co-UP
- it is unlikely to be NP-complete
- it is not known to be P