

# About Games and Trees

Barbara Jobstmann

Cadence Design Systems

Ecole Polytechnique Fédérale de Lausanne

Grenoble, December 2018

## Recap

Winning conditions are defined over Occ and Inf.

Occ( $\rho$ )	Inf( $\rho$ )
Reachability/Guarantee game	Büchi game
Safety game	co-Büchi game
Weak-parity game	Parity game
Obligation/Staiger-Wagner game	Muller game
LTL games	

## Recap

How did we solve those games?

Game	Solution
Reachability games	Attractor + Attractor Strategy
Safety games	like Reachability games
Büchi games	Recurrence set + Extended Attractor Strategy
co-Büchi games	like Büchi games
Weak-parity games	Alternation between $\text{Attr}_0$ and $\text{Attr}_1$
Obligation games	Reduction to Weak-parity games with record sets
Parity games	Recursive algorithm, Progress-Measure algorithm, and Strategy Improvement algorithm

# Muller, Rabin, and Streett Games

## Muller Games

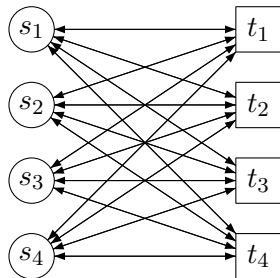
Given a game graph  $G = (S, S_0, E)$  and a Muller condition  $\mathcal{F} \subseteq \mathcal{P}(S)$ , then a play  $\rho$  is winning for Player 0 if exists  $F \in \mathcal{F}$  s.t.

$$\text{Inf}(\rho) = F.$$

Recall, in Staiger-Wagner games, we had  $\text{Occ}(\rho) = F$ .

## Example

Player 0 wins iff the number of states in  $S_0 = \{s_1, s_2, s_3, s_4\}$  visited infinitely often is equal to the lowest index of the states in  $S_1 = \{t_1, t_2, t_3, t_4\}$  visited infinitely often.



Winning condition in Muller form:  $F \in \mathcal{F}$  iff  $\min_i(t_i \in F) = |F \cap S_0|$ .

## Record the Past

For simplicity, we record only the  $s$ -states.

Visited letter	Record set
$s_1$	$s_1$
$s_3$	$s_1 s_3$
$s_3$	$s_1 s_3$
$s_4$	$s_1 s_3 s_4$
$s_2$	$s_1 s_2 s_3 s_4$
$s_4$	$s_1 s_2 s_3 s_4$
$s_3$	-"-
$s_4$	-"-
$s_4$	-"-

## Latest Appearance Record

Visited letter	Record set	LAR
$s_1$	$s_1$	$s_1 s_2 s_3 s_4 (1)$
$s_3$	$s_1 s_3$	$s_3 s_1 s_2 s_4 (3)$
$s_3$	$s_1 s_3$	$s_3 s_1 s_2 s_4 (1)$
$s_4$	$s_1 s_3 s_4$	$s_4 s_3 s_1 s_2 (4)$
$s_2$	$s_1 s_2 s_3 s_4$	$s_2 s_4 s_3 s_1 (4)$
$s_4$	$s_1 s_2 s_3 s_4$	$s_4 s_2 s_3 s_4 (2)$
$s_3$	-"-	..
$s_4$	-"-	..
$s_4$	-"-	..



## Example

Assume the states  $s_3$  and  $s_4$  are repeated infinitely often but not  $s_1, s_2$ . Then:

- ▶ the states  $s_1$  and  $s_2$  eventually arrive at the last two positions and are not touched any more, so finally the hit appears at most on positions 1 and 2
- ▶ position 2 is hit again and again; if only position 1 is hit from some point onwards, only the same letter would be chosen from there onwards (and not two states  $s_3$  and  $s_4$  as assumed)

## Example

LAR-strategy for Player 0:

During play update and use the LAR as follows:

- ▶ shift the letter of the current state to the front
- ▶ record the position from where the current letter was taken
- ▶ move to the state whose index is the current hit position

This is a finite-state winning strategy with  $n! \cdot n$  memory states if  $n$  letter states and  $n$  number states occur in the game graph.

## From Muller to Parity Games

### Theorem

*For a game  $(G, \phi)$  with  $G = (S, S_0, E)$  and Muller winning condition  $\phi$  (using the set  $\mathcal{F} \subseteq 2^S$ ), there is a game  $(G', \phi')$  with  $G' = (S', S'_0, E')$  and parity winning condition  $\phi'$  such that  $(G, \phi) \leq (G', \phi')$*

### Proof.

Assume  $n = \{1, \dots, n\}$ . Define  $S' := \text{LAR}(S)$

$\text{LAR}(S)$  is the set of pairs  $((i_1, \dots, i_n), h)$  consisting of a permutation of  $1, \dots, n$  and a number  $h \in \{1, \dots, n\}$ .

## Construction

Initialisation: For  $i \in S$  set

$$g(i) = ((i, i + 1, \dots, n, 1, \dots, i - 1), 1)$$

Definition of  $E'$ : Introduce an edge from  $((i_1 \dots i_n), h)$  to  $((i_m i_1 \dots i_{m-1} i_{m+1} \dots i_n), m)$  if  $(i_1, i_m) \in E$

## Construction

Initialisation: For  $i \in S$  set

$$g(i) = ((i, i + 1, \dots, n, 1, \dots, i - 1), 1)$$

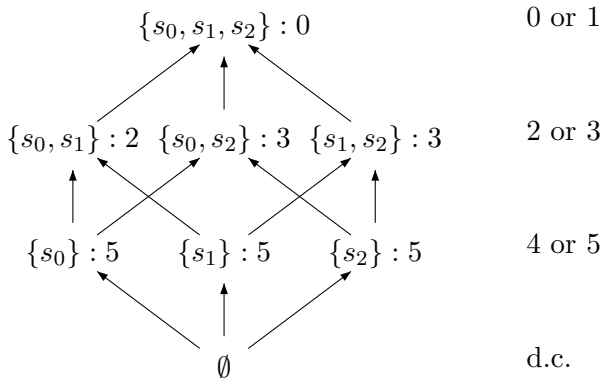
Definition of  $E'$ : Introduce an edge from  $((i_1 \dots i_n), h)$  to  $((i_m i_1 \dots i_{m-1} i_{m+1} \dots i_n), m)$  if  $(i_1, i_m) \in E$

How should we assign the priorities?

## Record Sets and Priorities

Recall, priorities in the reduction of Staiger-Wagner to Weak-Parity.

$$F = \{\{s_0, s_1\}, \{s_0, s_1, s_2\}\}.$$



## Construction(2)

Now, we are only interested in states visited infinitely often. The **hit value** tells as how many states are visited infinitely often.

E.g., if  $s_0$  and  $s_1$  are visited infinitely often, we see from some point on only the LARs:  $(s_0s_1 \dots, 1), (s_0s_1 \dots, 2), (s_1s_0 \dots, 1), (s_1s_0 \dots, 2)$ .

If  $\mathcal{F} = \{\{s_0, s_1\}\}$ , then we want plays that visit only  $(s_0s_1 \dots, 1)$  or  $(s_1s_0 \dots, 1)$  from some point on to be losing. So, the priorities signed to  $(s_0s_1 \dots, 2)$  or  $(s_0s_1 \dots, 2)$  need to override the priorities of  $(s_0s_1 \dots, 1)$  or  $(s_1s_0 \dots, 1)$ .

## Construction(2)

Now, we are only interested in states visited infinitely often. The **hit value** tells as how many states are visited infinitely often.

E.g., if  $s_0$  and  $s_1$  are visited infinitely often, we see from some point on only the LARs:  $(s_0s_1 \dots, 1), (s_0s_1 \dots, 2), (s_1s_0 \dots, 1), (s_1s_0 \dots, 2)$ . If  $\mathcal{F} = \{\{s_0, s_1\}\}$ , then we want plays that visit only  $(s_0s_1 \dots, 1)$  or  $(s_1s_0 \dots, 1)$  from some point on to be losing. So, the priorities signed to  $(s_0s_1 \dots, 2)$  or  $(s_0s_1 \dots, 2)$  need to override the priorities of  $(s_0s_1 \dots, 1)$  or  $(s_1s_0 \dots, 1)$ .

**Priorities p:**  $\text{LAR}(S) \rightarrow \{1, \dots, 2n\}$

$$p((i_1 \dots i_n, h)) = 2n - \begin{cases} 2h - 1 & \text{if } \{i_1 \dots i_n\} \notin \mathcal{F} \\ 2h & \text{if } \{i_1 \dots i_n\} \in \mathcal{F} \end{cases}$$



## Proof of Correctness

### Lemma

*Given a play  $\rho$  in  $(G, \phi)$  and its counterpart  $\rho'$  in  $(G', \phi')$ , then  $\text{Inf}(\rho) = F$  with  $|F| = m$  iff*

- 1. in  $\rho'$  the hit value is  $> m$  only finitely often*
- 2. in  $\rho'$  the hit-segment is equal to  $F$  infinitely often*

### Proof (forward).

Let  $\text{Inf}(\rho) = F$  and  $|F| = m$ . Choose  $k$  and  $k' > k$  s.t. for all  $j > k$   $\rho(j) \in F$  and  $\{\rho(k), \dots, \rho(k' - 1)\} = F$ .

By construction of  $\rho'$ , the  $F$ -states  $F = \{i_1, \dots, i_m\}$  are at the beginning of  $\rho'(k')$  and for every  $k'' > k'$  the hit is always  $\leq m$  (1).

## Proof of Correctness

### Proof (forward cont.)

For the hit equal to  $m$  the hit-segment must be the set  $F$ . So, for (2) it suffices to show that the hit is infinitely often equal to  $m$ . Assume the hit is only finitely often equal to  $m$ , then eventually the LAR-entries  $i_m, i_{m+1}, \dots, i_n$  are not changed anymore (and so, these states are not visited anymore). Then,  $|\text{Inf}(\rho)| < m$ , which contradicts  $\text{Inf}(\rho) = F$  with  $|F| = m$ .

### Proof (backwards).

Assume (1) and (2) holds. It follows from (1), that the LAR-entries  $i_{m+1}, \dots, i_n$  in  $\rho'$  are fixed from some point  $j_0$  onwards. So, the states  $i_{m+1}, \dots, i_n$  are not visited anymore after  $j_0$ . From, (2) it follows that  $i_{m+1}, \dots, i_n$  are not in  $F$  (i.e.,  $\text{Inf}(\rho) \subseteq F$ ).

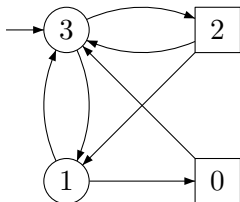
## Proof of Correctness

Proof (backwards cont.)

For  $F \subseteq \text{Inf}(\rho)$ , assume  $q \in F$  but  $q \notin \text{Inf}(\rho)$ .

Since  $q \in F$  and  $\text{hit-segment} = F$  infinitely often (2), we know that  $q \in \text{hit-segment}$  infinitely often. Furthermore, since  $|\text{hit-segment}| \leq m$  from some point on (1), it follows that from some point on the index  $i$  of  $q$  in the hit segment is  $\leq m$ . From  $q \notin \text{Inf}(\rho)$  it follows that from some point onwards  $q$  can only stay in the same position in the LAR or go to the right and its final position  $i$  is  $> m$ . Contradiction.

## Example



$$\rho \in \text{Win} \leftrightarrow \{0, 2\} \subseteq \text{Inf}(\rho)$$

$$\mathcal{F} = \{\{0, 2\}, \{0, 1, 2\}, \{0, 1, 2, 3\}\}$$

## Summary

We can solve Muller games by reduction to parity games using the **Last Appearance Record** construction.

## Summary

We can solve Muller games by reduction to parity games using the **Last Appearance Record** construction.

Finally, **Rabin and Streett** games can be viewed as Muller games.

## Back to Tree Automata

## Muller tree automaton

Recall, a Muller tree automaton over  $\Sigma$  is  $A = (S, s_0, T, \mathcal{F})$ , where

- ▶  $S$  is a finite set of states,
- ▶  $s_0 \in S$  is an initial state,
- ▶  $T : S \times \Sigma \rightarrow 2^{S \times S}$  is a transition function
- ▶  $\mathcal{F} \subseteq 2^S$  is the set of accepting sets.

Given an input tree  $t$ , a run  $\pi$  of  $A$  over  $t$  is **accepting** iff for every path  $\sigma$  in  $t$ :

$$\text{Inf}(\pi|_{\sigma}) \in \mathcal{F}$$



## Parity tree automaton

A Parity tree automaton over  $\Sigma$  is  $A = (S, s_0, T, p)$ , where

- ▶  $S$  is a finite set of states,
- ▶  $s_0 \in S$  is an initial state,
- ▶  $T : S \times \Sigma \rightarrow 2^{S \times S}$  is a transition function
- ▶  $p : S \rightarrow \{0, \dots, k\}$  is a priority function.

Given an input tree  $t$ , a run  $\pi$  of  $A$  over  $t$  is **accepting** iff for every path  $\sigma$  in  $t$ :

$$\min_{s \in \text{Inf}(\pi|_{\sigma})} p(s) \text{ is even}$$

## Example

A parity tree automaton over  $\Sigma = \{a, b\}$  that recognizes all binary trees

$$\mathcal{T} = \{t \in \mathcal{T}^\omega(\Sigma) \mid \text{each path through } t \text{ has only finitely many } b\}$$

- ▶  $S = \{q_a, q_b\}$
- ▶  $I = \{q_a, q_b\}$
- ▶  $T(q_a, a) = \{(q_a, q_a)\}$ ,  $T(q_b, a) = \{(q_a, q_a)\}$   
 $T(q_a, b) = \{(q_b, q_b)\}$ ,  $T(q_b, b) = \{(q_b, q_b)\}$
- ▶  $p(q_a) = 2$ ,  $p(q_b) = 1$

# Parity Automata $\leftrightarrow$ Muller Automata

## Theorem

1. *For any parity tree automaton one can construct an equivalent Muller tree automaton.*
2. *For any Muller tree automaton one can construct an equivalent parity tree automaton.*

## Proof 1.

Given a parity tree automaton  $A = (S, s_0, T, p)$  keep states and transitions and define  $\mathcal{F}$  as follows:

$$\mathcal{F} = \{F \in 2^S \mid \min_{s \in F} p(s) \text{ is even}\}$$

## Parity $\leftrightarrow$ Muller

### Proof 2.

Copy the simulation of Muller games by parity games. Given a Muller tree automaton with state set  $S$  use for the parity tree automaton the state set  $\text{LAR}(S)$  and define the transition according to the LAR update rule.

Allow transition

$$((s_1 \dots s_n, i), a, (s'_1 \dots s'_n, j), (s''_1 \dots s''_n, k))$$

for transition  $(s_1, a, s'_1, s''_1)$  of the Muller automaton, where

- ▶  $(s'_1 \dots s'_n, j)$  is the LAR update for a visit to  $s'_1$  and
- ▶  $(s''_1 \dots s''_n, k)$  is the LAR update for a visit to  $s''_1$ .

Define priorities as in the simulation of Muller games by parity games.

## Tree Automata and Games

With any parity tree automaton  $A = (S, s_0, T, p)$  over  $\Sigma$  and any input tree  $t \in \mathcal{T}^\omega(\Sigma)$ , we can associate a parity game between

- ▶ Player **Automaton** and
- ▶ Player **Pathfinder**

with proceeds as follows:

- ▶ First, Automaton picks a transition in  $T$  (from  $s_0$ ) which matches the labels of the root of  $t$
- ▶ Then Pathfinder decides on a direction (left or right) to proceed to a son of the root
- ▶ Then Automaton chooses again a transition for this node (and compatible with the first transition)
- ▶ Then Pathfinder reacts again by branching left or right...

## Tree Automata and Games

Such a play give a sequence of transitions (and hence a sequence of states in  $S$ ) built up along a path chosen by Pathfinder.

Automaton wins the play iff the sequence of states satisfies the parity condition.

Given a parity tree automaton  $A = (S, s_0, T, p)$  over  $\Sigma$  and an input tree  $t$ , the *game graph*  $G_{A,t} = (S_0 \cup S_1, S_0, E)$  is defined by

- ▶  $S_0 = \{(w, t(w), s) \mid w \in \{0, 1\}^*, t(w) \in \Sigma, s \in S_0\}$ ,
- ▶  $S_1 = \{(w, t(w), \tau) \mid w \in \{0, 1\}^*, t(w) \in \Sigma, \tau \in T\}$ ,

and the edges relation  $E$  is such that successive game positions are compatible with the transitions in  $A$  on  $t$ .

The priority of a triple  $u = (w, t(w), s)$  or  $(w, t(w), (s, t(w), s', s''))$  is the priority  $p(s)$ . (Standard initial position:  $(\epsilon, t(\epsilon), s_0)$ )

## Tree Automata and Games

### Lemma

*The tree automaton  $A$  accepts an input tree  $t$  iff in the parity game over  $G_{A,t}$  there is a winning strategy for player Automaton from the initial position  $(\epsilon, t(\epsilon), s_0)$ .*

### Proof.

A successful run of  $A$  on  $t$  yields a winning strategy for Automaton in the parity game over  $G_{A,t}$ : Along each path the suitable choice of transitions is fixed by the run.

Conversely, a winning strategy for Automaton over  $G_{A,t}$  clearly provides a method to build up a successful run of  $A$  on  $t$ . Just apply this winning strategy along arbitrary paths.

## Summary: Tree Automaton

- ▶ Tree Automata can be viewed as games between **Automaton** and **Pathfinder**
- ▶ Parity and Muller tree automata can be reduced to each other
- ▶ (Same holds for Rabin/Streett, Parity, and Muller tree automata)
- ▶ Radu showed closure properties of Muller tree automaton (union, intersection, projection)
- ▶ Missing: complementation



## Complementation of Parity Tree Automaton

We will show basic idea.

- ▶ To complement a given automaton  $A$  means to construct an automaton  $B$  s.t.

$$t \notin A \leftrightarrow t \in B$$

- ▶ Due to the run lemma, complementation means to conclude from the **non-existence** of a winning strategy of Player Automaton in  $G_{A,t}$  that there exists a winning strategy of Automaton in  $G_{B,t}$ .

Proof has two steps:

1. use determinacy of parity games to show that if Automaton has no winning strategy over  $G_{A,t}$ , then Pathfinder has a winning strategy over  $G_{A,t}$  (from  $(\epsilon, t(\epsilon), s_0)$ )
2. Convert Pathfinder's strategy into an Automaton strategy.

# Complementation of Parity Tree Automaton

## Theorem

*For any parity tree automaton  $A$  over  $\Sigma$ , one can construct a Muller tree automaton (and therefore a parity tree automaton)  $B$  over  $\Sigma$  that recognizes  $\mathcal{T}^\omega(\Sigma) \setminus \mathcal{L}(A)$*

## Proof.

From Step 1 (determinacy of parity games), we know there exists a (memoryless) winning strategy  $f : S_1 \rightarrow \{0, 1\}$  for Player Pathfinder.

$$f : \{0, 1\}^* \times \Sigma \times T \rightarrow \{0, 1\}$$

decompose  $f$  into a family of strategies parameterized by  $w \in \{0, 1\}^*$

$$f_w : \Sigma \times T \rightarrow \{0, 1\}$$

## Complementation of Parity Tree Automaton

Let  $I$  be the set of all possible **local instructions**  $i : \Sigma \times T \rightarrow \{0, 1\}$ .

Then,  $f$  can be represented as  $I$ -labeled binary tree  $s$  with  $s(w) = f_w$ .

Let  $s \cdot t$  be the corresponding  $(I \times \Sigma)$ -labeled tree

$$s \cdot t(w) = (s(w), t(w)) \text{ for } w \in \{0, 1\}^*.$$

Since  $f$  exists, we know there is an  $I$ -labeled tree  $s$  s.t. for all sequences  $\tau_0\tau_1 \dots$  of transitions chosen by Automaton and for all paths (in path for the unique)  $\pi \in \{0, 1\}^*$ , the generated state sequence violates the parity condition.

Intuitively,  $f$  tells the “new” automaton for every tree  $t \notin \mathcal{L}(A)$  which path to track for a given transition sequences in order to reject/accept the tree  $t$ .

# Complementation of Parity Tree Automaton

So, we know:

1. There exists an  $I$ -labeled tree  $s$  such that  $s \cdot t$  satisfies
2. for all  $\pi \in \{0, 1\}^\omega$
3. for all  $\tau_0 \tau_1 \dots \in T^\omega$
4. if the sequence  $s|_\pi$  of local instructions applied to the sequence of tree labels  $t|_\pi$  and the sequence  $\tau_0 \tau_1 \dots$  produces the path  $\pi$ , then the state sequence determined by  $\tau_0 \tau_1 \dots$  violates the parity condition.

## Complementation of Parity Tree Automaton

- ▶ Condition 4 is a property of  $\omega$ -words over  $I \times \Sigma \times T \times \{0, 1\}$ , which can be checked by a Muller word automaton  $M_4$ .
- ▶ Condition 3 is a property of  $\omega$ -words over  $I \times \Sigma \times \{0, 1\}$  checked by  $M_3$ , which results from  $M_4$  by universally quantifying  $T$  (negate, project, negate).
- ▶ Condition 2 is a property of  $(I \times \Sigma)$ -labeled trees, which can be checked by a Muller tree automaton  $M_2$  that simulates  $M_3$  along each path.
- ▶ Condition 1, apply nondeterminism, a Muller tree automaton  $B$  can be built by guessing a tree  $s$  on the input tree  $t$  and running  $M_2$  on  $s \cdot t$ .