Obligation and Reactivity Games, Tree Automata

Hierarchy

4. Reactivity: Muller, Parity

3. Recurrence: Büchi — Persistence: co-Büchi

2. Obligation: Staiger-Wagner, Weak-Parity

1. Reachability — Safety

Obligation Games

We consider games where the winning condition for Player 0 (on the play) is

- a Boolean combination of reachability conditions
- equivalently: a condition on the set Occ

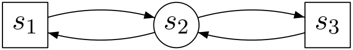
Standard form: Staiger-Wagner winning condition $\mathcal{F} = \{F_1, \ldots, F_k\}$

Player 0 wins play ρ iff $\mathrm{Occ}(\rho) \in \mathcal{F}$.

We call these games obligation games (or Staiger-Wagner games).

Example

$S = \{s_1, s_2, s_3\}$ $\mathcal{F} = \{\{s_1, s_2, s_3\}\}$



No winning strategy is positional.

There is a finite-state winning strategy.

Weak Parity Games

Method for solving Staiger-Wagner games:

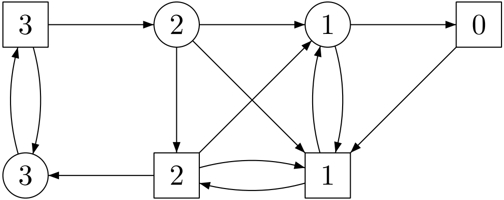
1. Solve weak parity games.
2. Reduce Staiger-Wagner games to weak parity games.

A weak parity game is a pair (G, p), where

- $G = (S, S_0, E)$ is a game graph and
- $p : S \to \{0, \ldots, k\}$ is a priority function mapping every state in S to a number in $\{0, \ldots, k\}$.

A play ρ is winning for Player 0 iff the minimum priority occurring in ρ is even: $\min_{s \in \mathrm{Occ}(\rho)} p(s)$ is even

Example



Weak Parity Games

Theorem

For a weak parity game one can compute the winning regions W_0, W_1 and also construct corresponding positional winning strategies.

Proof.

Let $G = (S, S_0, E)$ be a game graph, $p : S \to \{0, \ldots, k\}$ a priority function. Let $P_i = \{s \in S \mid p(s) = i\}$.

First steps if $P_0 \neq \emptyset$: We first compute $A_0 = \mathrm{Attr}_0(P_0)$, clearly from here Player 0 can win.

In the rest game, we compute $A_1 = \mathrm{Attr}_1(P_1 \setminus A_0)$ from here Player 1 can win.

General Construction

Aim: Compute A_0, $A_1, \ldots A_k$

Let G_i be the game graph restricted to $S \setminus (A_0 \cup \ldots A_{i-1})$.

$\mathrm{Attr}_0^{G_i}(M)$ is the 0-attractor of M in the subgraph induced by G_i

$$
\begin{aligned}
A_0 \quad & := \mathrm{Attr}_0(P_0) \\
A_1 \quad & := \mathrm{Attr}_1^{G_1}(P_1 \setminus A_0) \\
\text{for } i > 1 : & \\
A_i \quad & := \begin{cases} \mathrm{Attr}_0^{G_i}(P_i \setminus (A_0 \cup .. \cup A_{i-1})) & \text{if } i \text{ is even} \\ \mathrm{Attr}_1^{G_i}(P_i \setminus (A_0 \cup .. \cup A_{i-1)}) & \text{if } i \text{ is odd} \end{cases}
\end{aligned}
$$

Correctness

Correctness Claim:

$$W_0 = \bigcup_{i \text{ even}} A_i \text{ and } W_1 = \bigcup_{i \text{ odd}} A_i$$

and the union of the corresponding attractor strategies are positional winning strategies for the two players on their respective winning regions.

Prove by induction on $j = 0, \ldots, k$ the following:

$$\bigcup_{i=0..j, i \text{ even}} A_i \subseteq W_0 \text{ and } \bigcup_{i=1..j, i \text{ odd}} A_i \subseteq W_1$$

Correctness (cont.)

Base:

- i=0: $A_0 = \mathrm{Attr}_0(P_0) \subseteq W_0$
- i=1: $A_1 = \mathrm{Attr}_1^{G_1}(P_1 \setminus A_0) \subseteq W_1$

Induction step:

- i even: Consider play ρ starting A_i that complies to attractor strategy.
  - Case 1: ρ eventually leaves A_i to some A_j (from a Player-1 state), which $j < i$ and even, then Player 0 wins by induction hypothesis.
  - Case 2: ρ visits P_i, then we need to show that ρ visits only states with $p(s) \geq i$. Consider a state $s \in A_i$ that visits P_i, then
    - if $s \in S_0$, then not all edges lead to states with lower priority, otherwise $s \in A_j$ for some $j < i$. Contradiction.

Correctness (cont.)

- Case 2 (cont.):
  - if $s \in S_1$, then all edges lead to states with priority $\geq i$. Any edge to a lower priority must lead to A_j with even j (Case 1). If there were edges to states s' with priority $j < i$ and j odd, then s' would already be in A_j. Contradiction.
- i odd: switch players

Obligation/Staiger-Wagner to Weak-Parity Games

- How to translate a Staiger-Wagner automaton to Weak-Parity automaton?
- Idea: record visited states during a run
- Record set: $R \subseteq S$
- Question: How to give priorities?

Record Sets and Priorities

Assume automaton with states $\{s_0, s_1, s_2\}$.

Consider possible record sets:

$$
\begin{tikzcd}
 & \{s_0, s_1, s_2\} & \\
\{s_0, s_1\} \arrow[ur] & \{s_0, s_2\} \arrow[u] & \{s_1, s_2\} \arrow[ul] \\
\{s_0\} \arrow[u] \arrow[ur] & \{s_1\} \arrow[ul] \arrow[ur] & \{s_2\} \arrow[ul] \arrow[u] \\
 & \emptyset \arrow[ul] \arrow[u] \arrow[ur] &
\end{tikzcd}
$$

Assume the following run $s_1, s_0, s_1, s_0, s_2, \ldots$ and the acceptance condition $F = \{\{s_0, s_1\}, \{s_0, s_1, s_2\}\}$. How to assign priorities?

Record Sets and Priorities

$F = \{\{s_0, s_1\}, \{s_0, s_1, s_2\}\}$. How would you assign priorities?

$$
\begin{tikzcd}
 & \{s_0, s_1, s_2\} : 0 & & \text{0 or 1} \\
\{s_0, s_1\} : 2 \arrow[ur] & \{s_0, s_2\} : 3 \arrow[u] & \{s_1, s_2\} : 3 \arrow[ul] & \text{2 or 3} \\
\{s_0\} : 5 \arrow[u] \arrow[ur] & \{s_1\} : 5 \arrow[ul] \arrow[ur] & \{s_2\} : 5 \arrow[ul] \arrow[u] & \text{4 or 5} \\
 & \emptyset \arrow[ul] \arrow[u] \arrow[ur] & & \text{d.c.}
\end{tikzcd}
$$

From Staiger-Wagner to Weak Parity Automata

Given a deterministic Staiger-Wagner automaton $A = (S, I, T, F)$, we can construct an equivalent weak parity automaton $A' = (S', I', T', p)$ as follows:

$$
\begin{aligned}
S' &:= S \times 2^S \\
I' &:= (I, \{I\}) \\
T'((s, R), a) &:= (T(s, a), R \cup \{T(s, a)\} \\
p((s, R)) &:= 2 \cdot |S| - \begin{cases} 2 \cdot |R| & \text{if } R \in F \\ 2 \cdot |R| - 1 & \text{if } R \notin F \end{cases}
\end{aligned}
$$

Idea of Game Reduction

We want to solve Staiger-Wagner games. We use a reduction to weak parity games (and the positional winning strategies of weak parity games).

Reduction will transform a game (G, ϕ) into a game (G', ϕ') such that usually

- G' is (usually) larger than G
- ϕ' is simpler than ϕ (so the solution of (G', ϕ') is simpler than that of (G, ϕ))
- from a solution of (G', ϕ') we can construct a solution of (G, ϕ).

Concrete application: Transform Staiger-Wagner game into a weak parity game over a larger graph (from S proceed to $S \times 2^S$)

Game Reduction

Let $G = (S, S_0, E)$ and $G' = (S', S'_0, E')$ be game graphs with winning conditions ϕ and ϕ', respectively.

(G, ϕ) is reducible to (G', ϕ') if:

1. $S' = S \times M$ for a finite set M and $S'_0 = S_0 \times M$
2. Each play $\rho = s_0 s_1 \ldots$ over G is translated into a play $\rho' = s'_0 s'_1 \ldots$ over G' by
   - a function $g : S \to S \times M$ (marks the beginning of ρ').
   - for all states $(s, m) \in S \times M$ in G' and all states $s' \in S$ in G, if there exists an edge $(s, s') \in E$, then there is a unique m' with $((s, m), (s', m')) \in E'$
   - for each edge $((s, m), (s', m')) \in E'$ in G', there is an edge $(s, s') \in E$ in G
3. For all plays ρ and ρ' according to 2.: $\rho \in \phi$ iff $\rho' \in \phi'$

Application of Game Reduction

Theorem

Suppose (G, ϕ) is reducible to (G', ϕ') with extension set M, initial function g, and G and G' defined as before. Then, if Player 0 wins in (G', ϕ') from $g(s)$ with a memoryless winning strategy, then Player 0 wins in (G, ϕ) from s with a finite-state strategy.

Idea: Given a memoryless winning strategy $f : S'_0 \to S'$ from $g(s)$ for Player 0 in (G', ϕ'), we can construct a strategy automaton $A = (M, m_0, \delta, \lambda)$ for Player 0 in (G, ϕ).

Obligation/Staiger-Wagner Games

Theorem

Given a Staiger-Wagner game (G, ϕ), one can compute the winning regions of Player 0 and 1 and corresponding finite state strategies.

Proof.

We can apply game reduction with (G', ϕ') as follows:

$$
\begin{aligned}
G' &:= (S', S'_0, E') \\
S' &:= S \times 2^S \\
((s,R),(s',R')) \in E') &\quad \text{iff } (s,s') \in E, R' = R \cup \{s'\} \\
g(s) &= (s, \{s\}) \\
p((s,R)) &:= 2 \cdot |S| - \begin{cases} 2 \cdot |R| & \text{if } R \in \phi \\ 2 \cdot |R| - 1 & \text{if } R \notin \phi \end{cases}
\end{aligned}
$$

Hierarchy

4. Reactivity: Muller, Parity

3. Recurrence: Büchi   Persistence: co-Büchi

2. Obligation: Staiger-Wagner, Weak-Parity

1. Reachability   Safety

Parity Games

A Parity game is a pair (G, p), where

- $G = (S, S_0, E)$ is a game graph and
- $p : S \to \{0, \ldots, k\}$ is a priority function mapping every state in S to a number in $\{0, \ldots, k\}$.

A play ρ is winning for Player 0 iff the minimum priority visited infinitely often in ρ is even: $\min_{s \in \mathrm{Inf}(\rho)} p(s)$ is even.

Parity Games

Theorem

1. *Parity games are determined (i.e., each state belongs to* W_0 *or* W_1*), and the has a positional winning strategy.*
2. *Over finite graphs, the winning regions and winning strategies of the two players can be effectively computed.*

Overview

We will show two proofs:

- One for general (even infinite) game graph
- One constructive for finite game graphs to establish effectiveness.

Proof 1

Given $G = (S, S_0, E)$ with priority function $p : S \to \{0, \ldots, d\}$ and let $P_i = \{s \in S \mid p(s) = i\}$. We proceed by induction on the number of priorities.

- Base case: we either have an even or an odd priority

Proof 1

Given $G = (S, S_0, E)$ with priority function $p : S \to \{0, \ldots, d\}$ and let $P_i = \{s \in S \mid p(s) = i\}$. We proceed by induction on the number of priorities

- Base case: we either have an even or an odd priority
- Induction step: we assume that the minimum priority k is even (otherwise switch the roles of players 0 and 1 below).
  Let Π_1 be the set of vertices from which player 1 has a positional winning strategy.
  Show that from each vertex in $S \setminus \Pi_1$, player 0 has a positional winning strategy.

Proof 1: Induction step

Consider the subgame with vertex set $S \setminus \Pi_1$

- Case 1: $S \setminus \Pi_1$ does not reach the minimal priority k.

  Then, $S \setminus \Pi_1$ defines a subgame. Why?

  Induction hypothesis applies.

- Case 2: $S \setminus \Pi_1$ contains vertices of minimal (even) priority.

  Then, $S \setminus (\Pi_1 \cup \mathrm{Attr}_0(P_k \setminus \Pi_1))$ defines a subgame

Proof 1: Induction step

Player 0 can guarantee that starting from a vertex in $S \setminus \Pi_1$ the play remains there.

Either the play stays in $S \setminus (\Pi_1 \cup \mathrm{Attr}_0(P_k \setminus \Pi_1))$ from some point on, or it visits $\mathrm{Attr}_0(P_k \setminus \Pi_1)$ infinitely often.

In the first case player 0 wins by induction hypothesis with a positional strategy, in the second case by infinitely many visits to the lowest (even) priority, also with a positional strategy.

Altogether: Player 0 wins from each vertex in $S \setminus \Pi_1$ with a positional strategy.

Proof 2

Given $G = (S, S_0, E)$ with S finite and priority function $p : S \to \{0, \ldots, d\}$. We proceed by induction on the number of states denoted by n.

- Base case: we either have one Player-0 or Player-1 state with a selfloop (Note that every state in a game has at least one outgoing edge). Then the priority of the state determines if $S = W_0$ or $S = W_1$.
- Induction step: Let $P_i = \{s \mid p(s) = i\}$ be the set of states with priority i. Assume $P_0 \neq \emptyset$, otherwise assume $P_1 \neq \emptyset$ and switch the roles of Players 0 and 1 below. Finally, if $P_0 = P_1 = \emptyset$ decrease every priority by 2.

Proof (induction step cont.)

Choose $s \in P_0$ and let $X = \mathrm{Attr}_0(\{s\})$. Note that $S \setminus X$ is a subgame with $< n$ states.

The induction hypothesis gives a partition of $S \setminus X$ into winning regions U_0 and U_1 for Player 0 and 1, respectively, and corresponding positional winning strategies.

- Case 1: Player 0 can guarantee a transition from s to $U_0 \cup X$, i.e., if $s \in S_0$, then there exists $s' \in U_0 \cup X$ such that $(s, s') \in E$ or if $s \in S_1$, then for all $(s, s') \in E$, $s' \in U_0 \cup X$ holds.

  Claim:

  (i) $U_0 \cup X \subseteq W_0$

  (ii) $U_1 \subseteq W_1$.

Proof (Case 1 cont.)

The positional strategy for Player 0 on $U_0 \cup X$ is:

1. On U_0 play according to the positional strategy given by the induction hypothesis
2. On X $(= \mathrm{Attr}_0(\{s\}))$ play according to the attractor strategy. Then eventually reach s
3. From s "move back" to $U_0 \cup X$ (by the assumption of Case 1).

For Player 1 use the positional strategy on U_1 given by the induction hypothesis.

Proof of claim: (ii) is clear, since starting in U_1 Player 1 can guarantee that the play remains in U_1. For (i), the play remains in $U_0 \cup X$ if the strategy for state s is followed. If the play eventually remains in U_0, then Player 0 wins by induction hypothesis, otherwise the play passes through s infinitely often, which is winning as well.

Proof (Case 2)

- Case 2: Player 1 can guarantee a transition to U_1 from s, i.e., if $s \in S_0$, then all edges $(s, s') \in E$ lead to U_1 ($s' \in U_1$), and if $s \in S_1$, then there exists $s' \in U_1$ such that $(s, s') \in E$.

  Let $Y = \mathrm{Attr}_1(U_1)$, then $s \in Y$ and $S \setminus Y$ is a subgame with $< n$ states. The induction hypothesis gives winning region V_0 and V_1 and corresponding positional winning strategies.

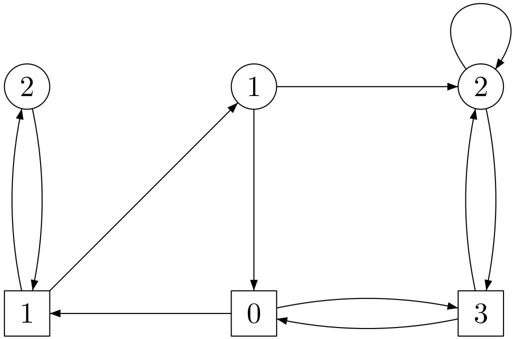
  Claim:

  (i) $V_0 \subseteq W_0$

  (ii) $V_1 \cup Y \subseteq W_1$.

  Proof of claim: (i) is clear, since Player 0 can guarantee to stay within V_0. For (ii), for all states in Y, Player 1 can guarantee to move to U_1 and stay there. From $t \in V_1$ Player 0 can either move to Y or stay in V_1. Both choices are winning for Player 1.

Example



Games and Tree Automata

Recap

Winning conditions are defined over Occ and Inf.

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

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 Attr_0 and Attr_1
Obligation games	Reduction to Weak-parity games + record sets
Parity games	Recursive algorithm

Muller Games

Muller Games

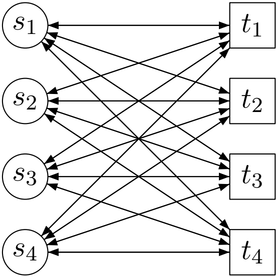
Given a game graph $G = (S, S_0, E)$ and a Muller condition $\mathcal{F} \subseteq \mathcal{P}(S)$, then a play ρ is winning for Player 0 if exists $F \in \mathcal{F}$ s.t.

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

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

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

Proof.

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

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

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

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

Construction

Initialisation: For $i \in S$ set

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

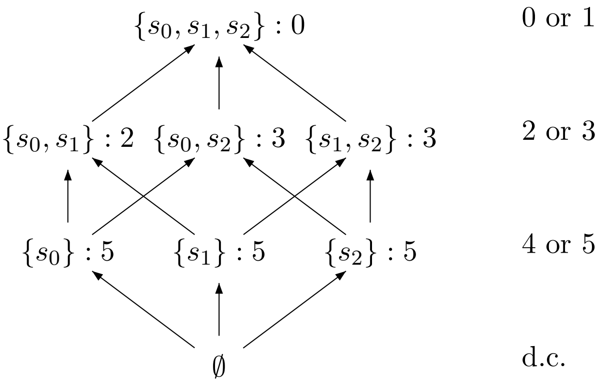
Definition of E': Introduce an edge from $((i_1 \ldots i_n), h)$ to $((i_m i_1 \ldots i_{m-1} i_{m+1} \ldots i_n), m)$ if $(i_1, i_m) \in E$

How should we assign the priorities?

Record Sets and Priorities

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

$F = \{\{s_0, s1\}, \{s0, s1, s2\}\}$.



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: $(s_0 s_1 \dots, 1)$,$(s_0 s_1 \dots, 2)$, $(s_1 s_0 \dots, 1)$, $(s_1 s_0 \dots, 2)$.

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

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: $(s_0 s_1 \ldots, 1)$,$(s_0 s_1 \ldots, 2)$, $(s_1 s_0 \ldots, 1)$, $(s_1 s_0 \ldots, 2)$.

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

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

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

Proof of Correctness

Lemma

Given a play ρ in (G, ϕ) and its counterpart ρ' in (G', ϕ'), then $Inf(\rho) = F$ with $|F| = m$ iff

1. *in ρ' the hit value is $> m$ only finitely often*
2. *in ρ' 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. forall $j > k$ $\rho(j) \in F$ and $\{\rho(k), \ldots, \rho(k'-1)\} = F$.
By construction of ρ', the F-states $F = \{i_1, \ldots, i_m\}$ 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, $|\mathrm{Inf}(\rho)| < m$, which contradicts $\mathrm{Inf}(\rho) = F$ 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 ρ' 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., $\mathrm{Inf}(\rho) \subseteq F$).

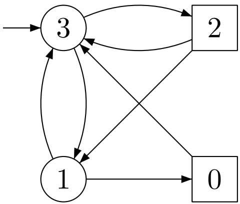
Proof of Correctness

Proof (backwards cont.)

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

Since $q \in F$ and hit-segment $= F$ infinitely often (2), we know that $q \in$ 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



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

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

Back to Tree Automata

Muller tree automaton

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

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

Given an input tree t, a run π of A over t is accepting iff for every path σ in t:

$$\mathrm{Inf}(\pi_{|\sigma}) \in \mathcal{F}$$

Parity tree automaton

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

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

Given an input tree t, a run π of A over t is accepting iff for every path σ in t:

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

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 $\mathrm{LAR}(S)$ 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 Σ and any input tree $t \in \mathcal{T}^\omega(\Sigma)$, we can associate a parity game between

- Player Automaton and
- Player Pathfinder

that 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 Σ 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)
- We 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 Σ, one can construct a Muller tree automaton (and therefore a parity tree automaton) B over Σ 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 $f : S_1 \to \{0, 1\}$ for Player Pathfinder.

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

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

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

Complementation of Parity Tree Automaton

Let I be the set of all possible local instructions $i : \Sigma \times T \to \{0, 1\}$.
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 $\tau_0\tau_1 \cdots \in T^\omega$
            4. if the sequence $s_{|\pi}$ of local instructions applied to the sequence of tree labels $t_{|\pi}$ and the sequence $\tau_0\tau_1 \ldots$ produces the path π, then the state sequence determined by $\tau_0\tau_1 \ldots$ violates the parity condition.

Complementation of Parity Tree Automaton

- Condition 4 is a property of ω-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 ω-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 $s \cdot t$.