# Formal Methods - Homework

## Closure under complementation of tree automata

Mikhail Dubov

January 21, 2016

# Outline

# Infinite trees & Tree automata

Automata considered so far are **word automata**:

- consume sequences of alphabet **symbols**
- $\omega$-automata consume **infinite sequences**, i.e. $\omega$-words

We will take a look at **tree automata** [1]:

- process **infinite trees**, not words
- are still **finite-state**

# Infinite trees & Tree automata: Motivation

Automata on infinite objects, in particular trees, play an important role in computer science:

- They allow to model **nonterminating systems**
- Tree automata are more suitable than words when **nondeterminism** needs to be modelled
- There are connections to logical theories (e.g. to **MSO** [2])

# Infinite trees: Definitions

**Infinite binary tree:**

- Set $T^\omega = \{0,1\}^*$ of all finite words on $\{0,1\}$
- Elements $u \in T^\omega$ are nodes:
  - $\varepsilon$ is the root
  - $u0$ and $u1$ are left and right successors of $u$
- $v$ is a successor of $u$ if there exists a $w \in T^\omega$ s.t. $v = uw$

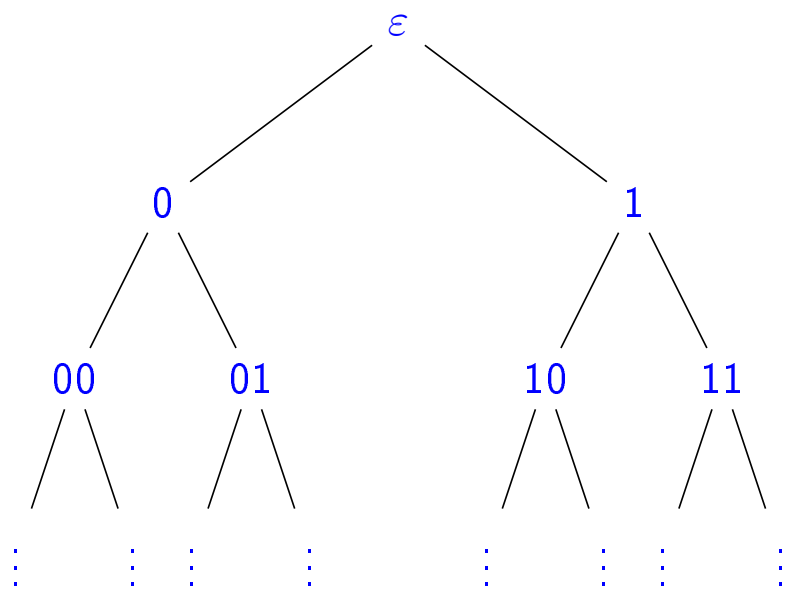



Figure: Infinite binary tree

# Infinite trees: Definitions

## $\Sigma$-labeled tree:

- Mapping $t : T^\omega \to \Sigma$ labels the nodes with symbols from $\Sigma$
- $T_\Sigma$ – set of all $\Sigma$-labeled trees
- $\omega$-word $\pi \in \{0,1\}^\omega$ is a **path** in the binary tree $T^\omega$
  - Prefixes of $\pi$ are nodes on that path
  - We are interested in the **labeling of a path** $\pi$ through $t$: $t|\pi$

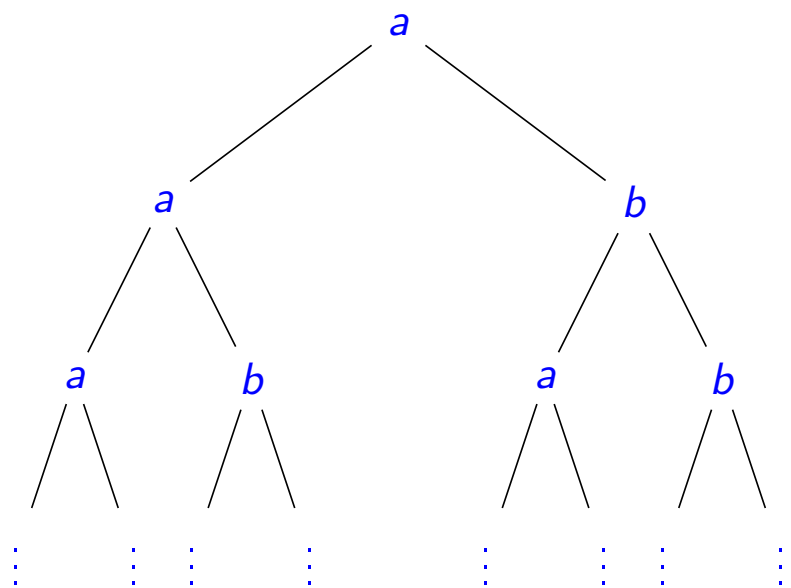



Figure: $\{a, b\}$-labeling: $t(\varepsilon) = a$, $t(w0) = a$, $t(w1) = b$

# Tree automata: Definitions

**Notdeterministic finite-state tree automaton:**

- Intuitive definition:
  - **Word automata:** each input symbol has one successor $\Longrightarrow$ word automaton enters **one** successor state
  - **Tree automata:** each input node has two successors $\Longrightarrow$ tree automaton enters **two** successor states
- Formal definition (**Muller tree automaton**):
  - Muller tree automaton is a quintuple $\mathcal{A} = (Q, \Sigma, \Delta, q_I, \mathcal{F})$
    - $Q$: finite state set
    - $\Sigma$: finite alphabet
    - $\Delta \subseteq Q \times \Sigma \times Q \times Q$: transition relation
    - $q_I$: initial state
    - $\mathcal{F}$: collection of sets of accepting states

# Tree automata: Definitions

- A **run** of $\mathcal{A}$ on an input **labeled tree** $t \in T_\Sigma$ is a **labeled tree** $\varrho \in T_Q$, satisfying:
  - $\varrho(\varepsilon) = q_I$
  - for all $w \in \{0,1\}^* : (\varrho(w), t(w), \varrho(w0), \varrho(w1)) \in \Delta$
- A run is called **successful** if for each path $\pi \in \{0,1\}^\omega$ the **Muller acceptance condition** is satisfied (i.e. $Inf(\varrho|\pi) \in \mathcal{F}$)
- $\mathcal{A}$ **accepts** the tree $t$ if there is a successful run of $\mathcal{A}$ on $t$
- The **tree language** recognized by $\mathcal{A}$ is the set $T(\mathcal{A}) = \{t \in T_A | \mathcal{A} \text{ accepts } t\}$

# Tree automata: Example

Consider the tree language
$T = \{t \in T_{\{a,b\}} |$ there is a path $\pi$ through $t$ such that $t|\pi \in (a+b)^*(ab)^\omega\}$.

Muller tree automaton $\mathcal{A}$ that recognizes $T$:

- "guesses" a path through $t$
- checks if the label of this path belongs to $(a+b)^*(ab)^\omega$
  - for this purpose, $\mathcal{A}$ has to memorize in its state the last read input symbol
  - we use states $q_a$ and $q_b$
  - therefore $\mathcal{F}$ includes the acceptance set $\{q_a, q_b\}$
  - for the nodes outside the "guessed path", we will use state $q_d$

# Tree automata: Example

$T = \{t \in T_{\{a,b\}} |$ there is a path $\pi$ through $t$ such that $t|\pi \in (a+b)^*(ab)^\omega\}$

Formally, $\mathcal{A} = (\{q_I, q_a, q_b, q_d\}, \{a, b\}, \Delta, q_I, \{\{q_a, q_b\}, \{q_d\}\})$.

Transition relation $\Delta$ includes:

- initial transitions:
  $(q_I, a, q_a, q_d), (q_I, a, q_d, q_a), (q_I, b, q_b, q_d), (q_I, b, q_d, q_b)$
- for paths we are not interested in:
  $(q_d, a, q_d, q_d), (q_d, b, q_d, q_d)$
- for checking $(ab)^\omega$:
  $(q_a, b, q_b, q_d), (q_a, b, q_d, q_b), (q_b, a, q_a, q_d), (q_b, a, q_d, q_a)$
- for falling back to checking the prefix $(a+b)^*$:
  $(q_a, a, q_I, q_d), (q_a, a, q_d, q_I), (q_b, b, q_I, q_d), (q_b, b, q_d, q_I)$

# Tree automata: Example

Example of a **successful run** $\varrho$ on the input tree $t$ from the previous example:

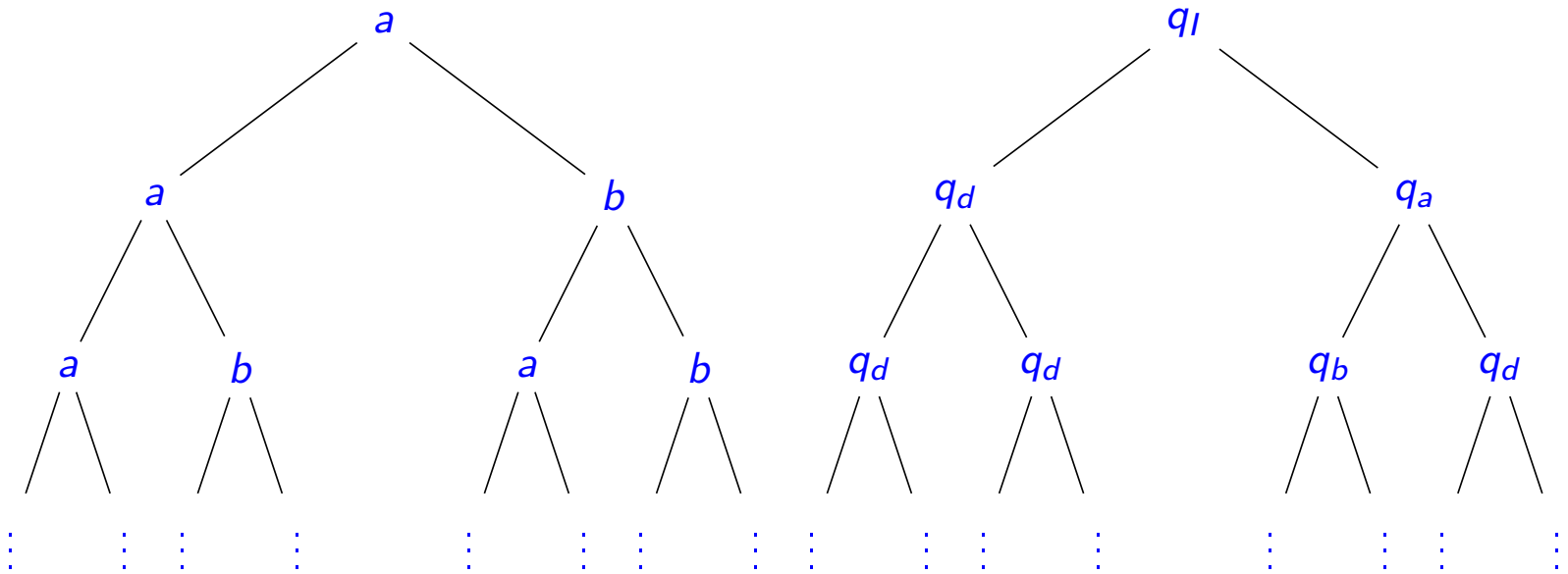



Figure: Input tree $t$ and a sample successful run $\varrho$ on this tree

# Tree automata: More formalisms

Exactly as with $\omega$-automata, we can define tree automata in different ways based on the **acceptance condition**.

- **Muller condition:** Set of states visited infinitely often is equal to one of the sets in $\mathcal{F}$
- **Parity condition:** For some coloring of the states, the minimal number of colors visited infinitely often is even
- **Rabin condition**, **Streett condition**, ...
- **Büchi condition:** For some coloring of the states and a set of accepting colors $F$, some of the colors visited infinitely often belong to $F$

*NB*: **Muller**, **parity**, **Rabin**, **Streett** tree automata are equivalent.
*NB*: **Büchi** tree automata are strictly weaker.

# Tree automata: More formalisms

**Parity tree automaton:**

- A quintuple $\mathcal{A} = (Q, \Sigma, \Delta, q_I, c)$
- $c : Q \to \{0, \ldots, k\}, k \in \mathbb{N}$ – coloring function
- A **run** of $\mathcal{A}$ on an input **labeled tree** $t \in T_\Sigma$ is again a **labeled tree** $\varrho \in T_Q$
- A run is **successful** if for each path $\pi \in \{0,1\}^\omega$ the **parity acceptance condition** is satisfied:
  $\min\{c(q) | q \in Inf(\varrho|\pi)\}$ is even

# Tree automata: More formalisms

Consider the tree language
$T = \{t \in T_{\{a,b\}} |$ for each path $\pi$ through $t$ holds $t|\pi \in a^\omega \cup (a+b)^* b^\omega\}$.

- Unlike the previous example, the automaton doesn't have to "guess" a path but should check all paths simultaneously
- Therefore, for each state the left and right successor states will be identical
- We can build a parity tree automaton for this language

# Parity tree automata: Example

$T = \{t \in T_{\{a,b\}} |$ for each path $\pi$ through $t$ holds $t|\pi \in a^\omega \cup (a+b)^* b^\omega\}$.

- Now we need only 3 states $q_I$, $q_a$, $q_b$ (no need for $q_d$)
- Transitions include:
  - initial transitions: $(q_I, b, q_b, q_b), (q_I, a, q_I, q_I)$ (reading $b$ means that $a^\omega$ is impossible)
  - in state $q_b$: $(q_b, b, q_b, q_b), (q_b, a, q_a, q_a)$
  - in state $q_a$: $(q_a, a, q_a, q_a), (q_a, b, q_b, q_b)$
- Coloring: $c(q_a) = 1$, $c(q_b) = c(q_I) = 2$
- For this coloring, the parity condition indeed ensures that $\mathcal{A}$ accepts $T$

# Tree automata languages

Finite tree automata languages are closed under union, intersection, projection and complementation.

**Complementation** is the most difficult one to prove:

- Original proof by Rabin [2] is rather complicated
- The proof can be much simplified using a **game-theoretically based** approach [4]

# Infinite games: Definitions

We are interested in **infinite two-person games** on **directed graphs**.

**Game = Arena + Winning Condition**

An **arena** is a triple $\mathcal{A} = (V_0, V_1, E)$, where

- $V_0$ is a set of **0-vertices** that belong to **Player 0**
- $V_1$ is a set of **1-vertices** that belong to **Player 1**
- $V = V_0 \cup V_1$, $V_0 \cap V_1 = \emptyset$
- $E \subseteq V \times V$ is a set of moves (edges)
- The set of **successors** of $v \in V$ is $vE = \{v' \in V | (v, v') \in E\}$

# Infinite games: Definitions

A **play** of a game goes as follows:

- A **token** is placed on some initial vertex $v \in V$
- If $v$ is a 0-vertex, then Player 0 moves the token from $v$ to a successor $v' \in vE$ of $v$
- If $v$ is a 1-vertex, then Player 1 moves the token from $v$ to a successor $v' \in vE$ of $v$
- $(V, E)$ is not required to be a bipartite graph so it is possible that Player 0(1) moves the token to a 0(1)-vertex
- The play can be **infinite**: it corresponds then to a **path** $\pi = v_0 v_1 v_2 \cdots \in V^\omega$ with $v_{i+1} \in v_i E \ \forall i \in \omega$
- The play can **stop** when a **dead end** (a vertex without successors) is reached: it corresponds then to a **path** $\pi = v_0 v_1 \dots v_l \in V^+$ with $v_{i+1} \in v_i E \ \forall i < l, v_l E = \emptyset$

# Infinite games: Definitions

A **game** is a pair $\mathcal{G} = (\mathcal{A}, Win)$, where:

- $\mathcal{A}$ is the **arena** of the game
- $Win \subseteq V^\omega$ is its **winning set** (the set of winning paths)

**Player 0** is the **winner** of a play $\pi$ of a game $\mathcal{G}$ iff

- $\pi$ is a finite play $\pi = v_0 v_1 \dots v_l \in V^+$ and $v_l$ is a dead-end 1-vertex (so Player 1 can't move anymore)
- $\pi$ is an infinite play and $\pi \in Win$

**Player 1** wins $\pi$ if Player 0 does not win $\pi$.

# Infinite games: Definitions

To ensure that our state space is finite (this is required for acceptance conditions we will consider next), we **color** the vertices of an arena:

- **Coloring function** $\chi : V \to C$ ($C$ is a finite set of colors)
- $\chi(\pi) = \chi(v_0)\chi(v_1)\chi(v_2)\ldots$ is the coloring of a play

# Infinite games: Acceptance conditions

We are interested in **winning sets** that can be described using the same acceptance conditions:

- **Muller condition:** $\pi \in Win$ iff $Inf(\chi(\pi)) \in \mathcal{F}$
  ($\mathcal{F}$ is a collection of sets of accepting colors)
- **Parity conditions:**
  - *max-parity condition:* $\pi \in Win$ iff $max(Inf(\chi(\pi))$ is even
  - *min-parity condition:* $\pi \in Win$ iff $min(Inf(\chi(\pi))$ is even
- **Büchi condition:** $\pi \in Win$ iff $Inf(\chi(\pi)) \cap F \neq \emptyset$
  ($F \subseteq C$ is a set of accepting colors)
- **Rabin condition**, **Streett condition**, ...

# Infinite games: Example

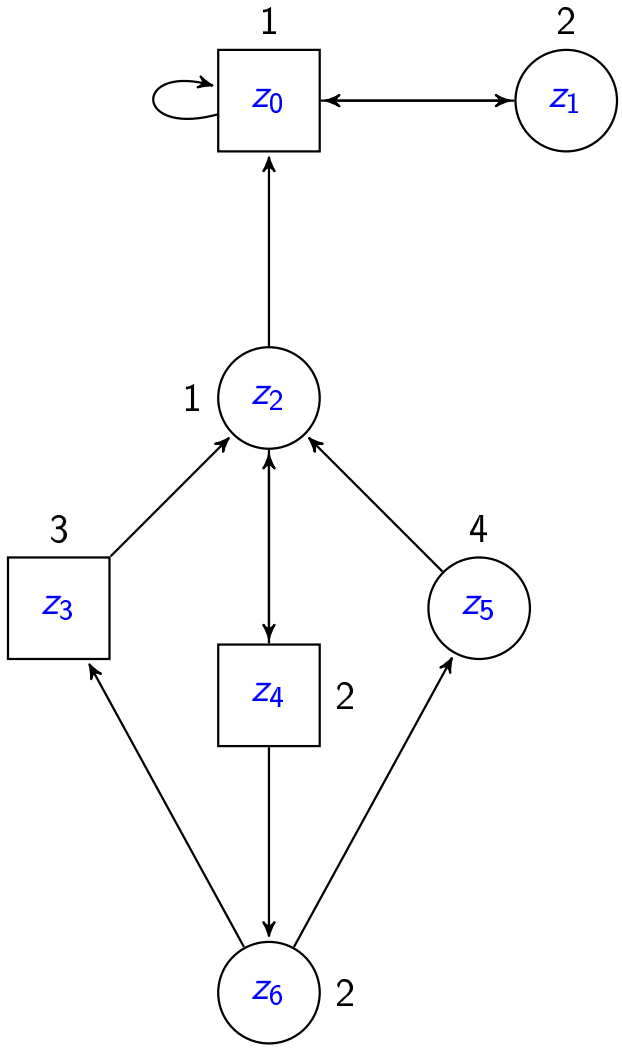



Figure: Example of a colored arena

# Infinite games: Example

In the example, $\mathcal{A} = (V_0, V_1, E, \chi)$ is a **colored arena** with:

- 0-vertices $V_0 = \{z_1, z_2, z_5, z_6\}$
- 1-vertices $V_1 = \{z_0, z_3, z_4\}$
- Edge relation $E = \{(z_0, z_1), (z_2, z_0), \dots\}$
- Colors $C = \{1, 2, 3, 4\}$
- Coloring funciton $\chi$: $\chi(z_4) = 2, \chi(z_0) = 1, \dots$

With the **Muller acceptance condition** given by $\mathcal{F} = \{\{1, 2\}, \{1, 2, 3, 4\}\}$,

- $\pi = z_6 z_3 z_2 z_4 z_2 z_4 z_6 z_5 (z_2 z_4)^\omega$ is a winning infinite play for Player 0, because
  - $\chi(\pi) = 23121224(12)^\omega$
  - $Inf(\chi(\pi)) = \{1, 2\} \in \mathcal{F}$
- $\pi = (z_2 z_4 z_6 z_3)^\omega$ is a winning infinite play for Player 1, because
  - $\chi(\pi) = (1223)^\omega$
  - $Inf(\chi(\pi)) = \{1, 2, 3\} \notin \mathcal{F}$

# Infinite games: Strategies

A **winning strategy** for Player 0 on $U \subseteq V$ is a function $f_0 : V^* V_0 \to V$ that

- is defined on every prefix of a play $\pi = v_0 v_1 \dots v_l$ that is conform with it (for every $i$ s.t. $0 \geq i < l$ and $v_i \in V_0$ the function $f_0$ is defined at $v_0 \dots v_i$ and $v_{i+1} = f_0(v_0 \dots v_i)$)
- leads to a **win** for Player 0 when applied repeatedly on a play starting in a vertex from $U$

$f_1$ for Player 1 is defined in the same way.

A winning strategy for Player 0 on $U = z_2, z_3, z_4, z_5, z_6$ is

$$f(x) = \begin{cases} z_4, & \text{if } \pi \in V^* z_2 \\ z_3, & \text{if } \pi \in V^* z_5 z_2 z_4 (z_2 z_4)^* z_6 \\ z_5, & \text{if } \pi \in V^* z_3 z_2 z_4 (z_2 z_4)^* z_6 \\ z_3, & \text{if } \pi \in (V \setminus \{z_3, z_5\})^* z_6 \end{cases}$$

# Infinite games: Strategies

A **strategy** is:

- **forgetful** if a finite amount of memory is sufficient for the corresponding Player to carry out this strategy
- **memoryless** if no memory is needed

In our example, the winning strategy for Player 0 is **forgetful**:

- Player 0 has to alternate between $z_3$ and $z_5$ to win from $z_6$
  $\Longrightarrow$ he has to remember whether he moved to $z_3$ or $z_5$ when the token was on $z_6$ last time

The winning strategy for Player 1 is **memoryless**:

- His strategy is just to move the token to $z_0$ when it is in $z_0$

# Game-theoretical view on tree automata

**Tree automata** and **infinite games** are **related**.

We can identify a parity tree automaton $\mathcal{A} = (Q, \Sigma, \Delta, q_I, c)$ and an input tree $t$ with an **infinite two-person game** $\mathcal{G}_{\mathcal{A},t}$:

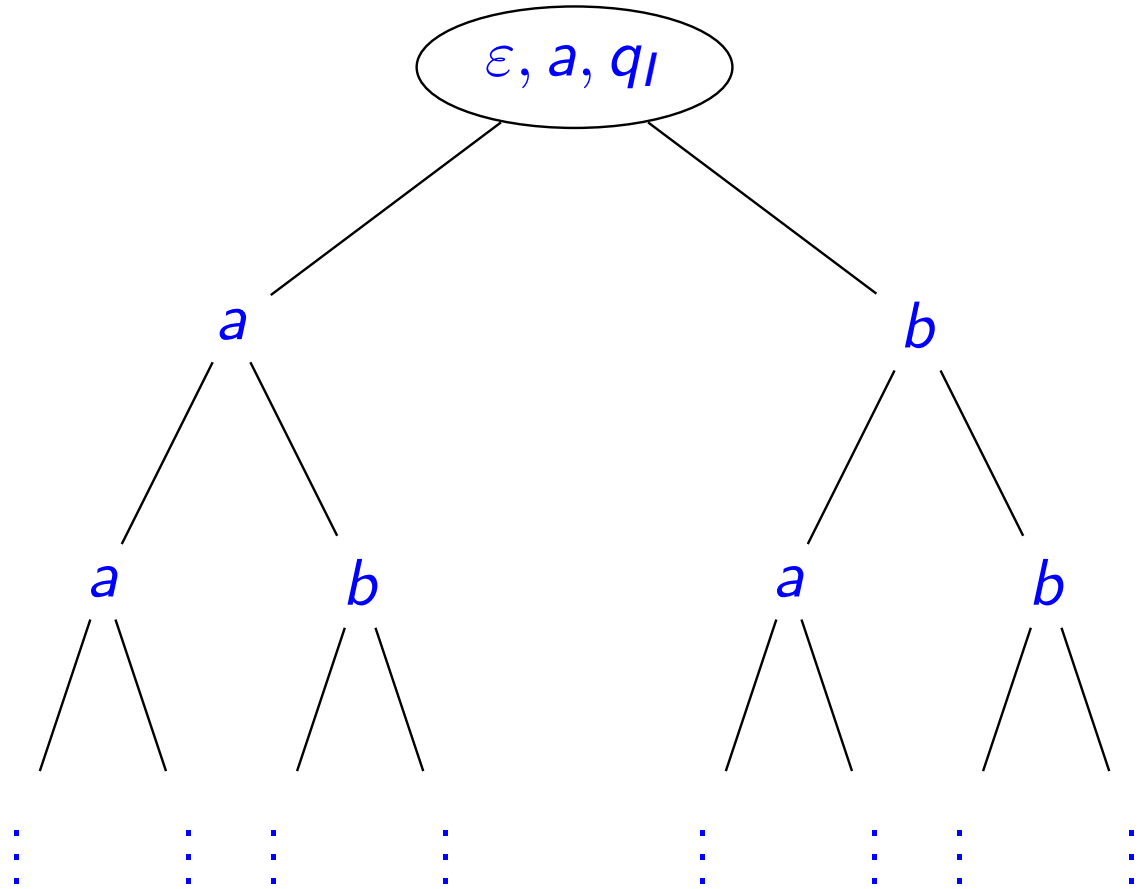
- Players 0 and 1 play the game on $t$, starting in the root
- The Players always **move alternately**:
  - Player 0 is an *"automaton"*: he picks a transition from $\Delta$ matching the symbol at the current node $t$
  - Player 1 is a *"pathfinder"*: he determines whether to procees with the left or the right successor.

This sequence of actions represents a **play** of the game and induces an infinite **sequence of states** visited along the path in $t$.

Player 0 **wins** if this infinite state sequence satisfies the acceptance condition of $\mathcal{A}$.

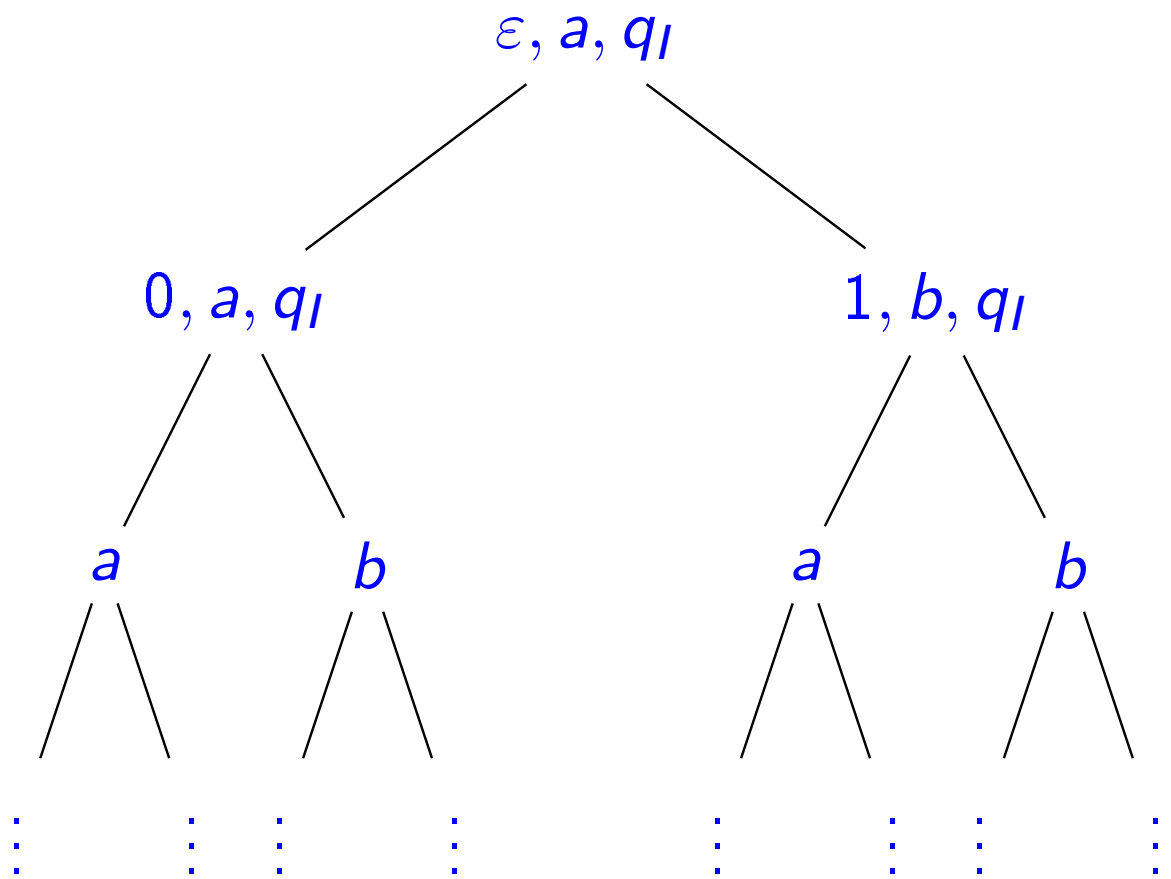
# Game on a tree: Example

Initial tree (from the previous example):



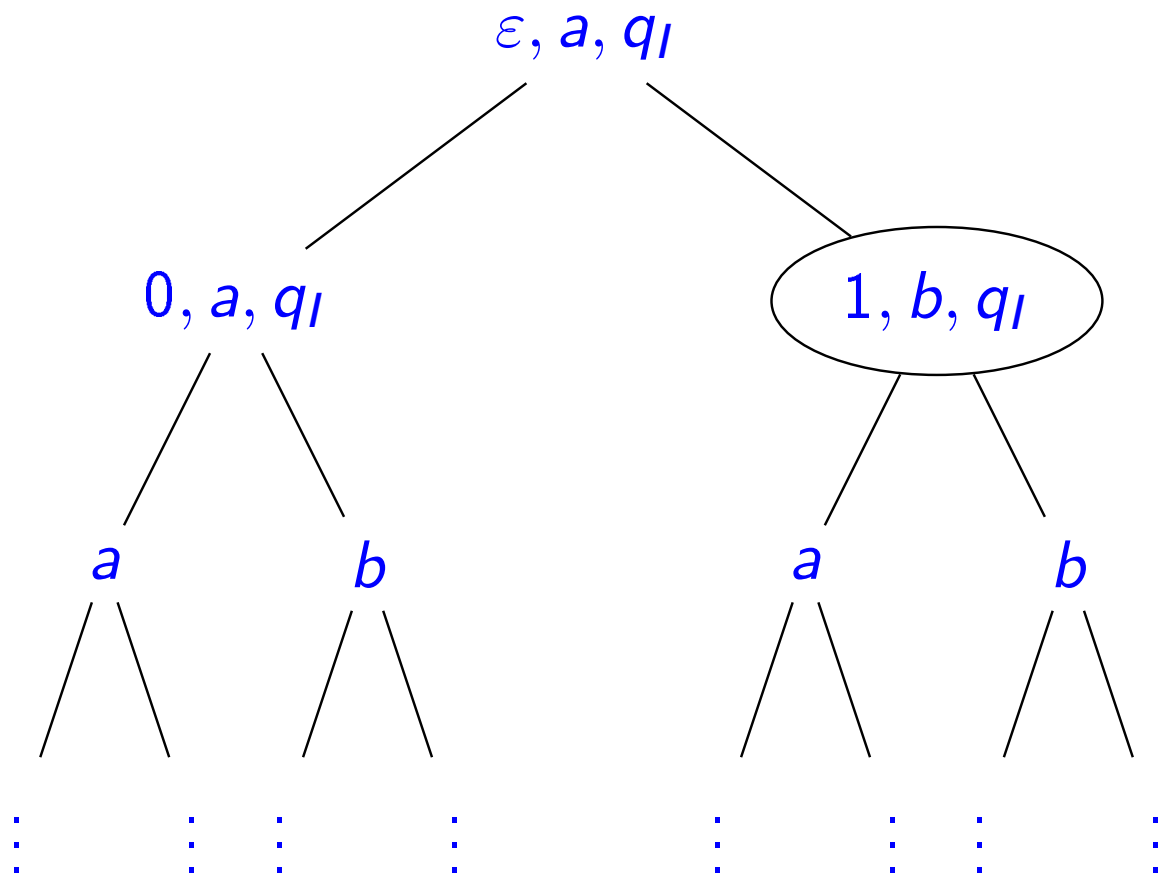
# Game on a tree: Example

**Player 0** picks a transition (the parity tree automaton is as before):



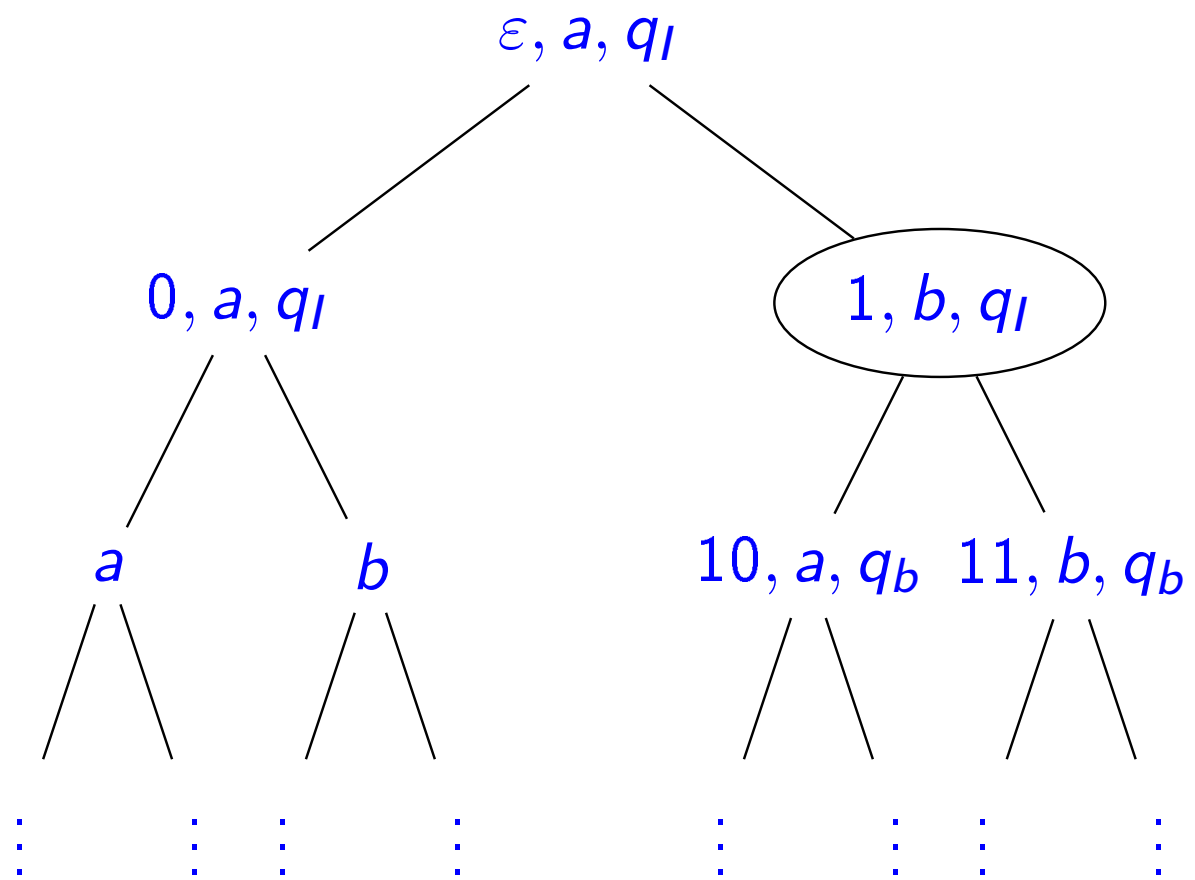
# Game on a tree: Example

**Player 1** picks the successor to proceed with:



# Game on a tree: Example

**Player 0** picks a transition:



# Game-theoretical view on tree automata

A few more remarks:

- Game positions for Player 0 are $V_0 = \{(w, q) | w \in \{0,1\}^*, q \in Q\}$
- Game positions for Player 1 are $V_1 = \{(w, \tau) | w \in \{0,1\}^*, \tau \in \Delta_{t(w)}\}$
- At each move, Players 0 and 1 choose the next position for each other
- We can color both position types by the colors of their states: $c((w, q)) = c(q)$ $c((w, (q, t(w), q'_0, q'_1))) = c(q)$
- Then $\mathcal{G}_{\mathcal{A},t}$ is a **parity game**

**Lemma 1**

*A tree automaton $\mathcal{A}$ accepts an input tree $t$ iff there is a winning strategy for Player 0 from position $(\varepsilon, q_I)$ in the game $\mathcal{G}_{\mathcal{A},t}$.*

# Complementation theorem

**Game-theoretical approach** enables a simpler proof for the **complementation theorem**.

**Theorem 2 (Complementation theorem)**

*The class of languages recognized by finite-state tree automata is closed under complementation.*

- **Problem:** Given a parity tree automaton $\mathcal{A}$, specify a tree automaton $\mathcal{B}$ that accepts all input trees rejected by $\mathcal{A}$
- **Game-theoretical approach:** there should be no winning strategy for Player 0 from position $(\varepsilon, q_I)$ in the game $\mathcal{G}_{\mathcal{A},t}$ (Lemma 1)
  $\Longrightarrow$ there exists a memoryless winning strategy starting at $(\varepsilon, q_I)$ for Player 1 (known from the theory of parity games)
- $\mathcal{B}$ will check exactly this

# Complementation theorem: Proof

**Memoryless strategy** for Player 1 is a function
$f : \{0,1\}^* \times \Delta \to \{0,1\}$,
determining a direciton 0 (left successor) or 1 (right successor):

- Corresponds to another function $\{0,1\}^* \to (\Delta \to \{0,1\})$
- This function essentially represents a **tree**
- We call such trees **strategy trees**; if the strategy is winning for Player 1 in $\mathcal{G}_{\mathcal{A},t}$, we call it a **winning tree**
- There exists a winning tree for Player 1 iff $\mathcal{A}$ (the parity tree automaton) does not accept the input tree $t$

# Complementation theorem: Proof

Given a parity tree automaton $\mathcal{A}$ and an input $t$ we decide whether a tree $s$ is not a winning tree for $t$ using an $\omega$-automaton $\mathcal{M}$ with parity acceptance condition:

- $\mathcal{M}$ checks for each path $\pi$ of $t$ and possible move by Player 0 whether the acceptane condition of $\mathcal{A}$ is met
- If at least once $\mathcal{A}$'s acceptance condition is met, $s$ cannot be a winning tree for $t$, and vice versa
- $\mathcal{M}$ handles $\omega$-words of form $u = (s(\varepsilon), t(\varepsilon), \pi_1)(s(\pi_1), t(\pi_1), \pi_2)\ldots$
- We denote the language of all these words $L(s, t)$
- E.g. for a path $\pi = 01100\ldots$ through the tree $t$, an $\omega$-word $u \in L(s, t)$ could look like $(f_\varepsilon, t(\varepsilon), 0)(f_0, t(0), 1)(f_{01}, t(01), 1)(f_{011}, t(011), 0)\ldots$

# Complementation theorem: Proof

We have:

- Tree automaton $\mathcal{A}$
- $\omega$-automaton $\mathcal{M} = (Q, \Sigma', \Lambda, q_i, c)$ handling any trees $s$ and $t$
- $\mathcal{M}$ nondeterministically checks for each possible move of Player 0 if the outcome is winning for Player 0
- $\mathcal{M}$'s alphabet
  $\Sigma' = \{(f, a, i) | f : \Delta \to \{0, 1\}, a \in \Sigma, i \in \{0, 1\}\}$
- $\mathcal{A}$ and $\mathcal{M}$ have the same acceptance condition

# Complementation theorem: Proof

**Lemma 3**

*The tree $s$ is a winning tree for $t$ if and only if $L(s,t) \cap L(\mathcal{M}) = \emptyset$*

$\Longrightarrow$ :

- Consider any play of the game $\mathcal{G}_{\mathcal{A},t}$
- Assume $(q_j, t(\pi_1 \dots \pi_j), q'_0, q'_1) \in \Delta$ to be the choice of Player 0 in node $\pi_1 \dots \pi_j$
- Player 1 plays according to $s$: the successor state is determined by $s(\pi_1 \dots \pi_j)$, i.e. $q_{j+1} \in \{q'_0, q'_1\}$
- $\varrho = q_I q_1 q_2 \dots$ is an infinite sequence of states visited along the play; it is also the run of $\mathcal{M}$ on the corresponding $\omega$-word $u = (s(\varepsilon), t(\varepsilon), \pi_1)(s(\pi_1), t(\pi_1), \pi_2) \cdots \in L(s,t)$
- We have $L(s,t) \cap L(\mathcal{M}) = \emptyset \implies \varrho$ is not accepting $\implies \mathcal{A}$ does not accept $t \implies s$ is a winning tree

# Complementation theorem: Proof

$\Longleftarrow$ :

- Assume $L(s,t) \cap L(\mathcal{M}) \neq \emptyset$, so there exists a path $\pi = \pi_1 \pi_2 \ldots$ s.t. the corresponding $\omega$-word $u = (s(\varepsilon), t(\varepsilon), \pi_1)(s(\pi_1), t(\pi_1), \pi_2) \cdots \in L(\mathcal{M})$
- So there is a successful run $\varrho = q_I q_1 q_2 \ldots$ of $\mathcal{M}$ on $u$.
- For each transition $(q_j, (s(\pi_1 \ldots \pi_j), t(\pi_1 \ldots \pi_j), \pi_{j+1}), q_{j+1})$ in $\varrho$, there is a corresponding transition $\tau_j = (q_j, t(\pi_1 \ldots \pi_j), q'_0, q'_1)$ of $\mathcal{A}$ such that $s(\pi_1 \ldots \pi_j) = f_{\pi_1 \ldots \pi_j}$ where $f_{\pi_1 \ldots \pi_j}(\tau_j) = \pi_{j+1}$
- If $\pi_{j+1} = 0$ then $q_{j+1} = q'_0$ otherwise $q_{j+1} = q'_1$
- Let these transitions $\tau_j$ be the choices of Player 0 (Player 1 reacts by choosing $s(\pi_1 \ldots \pi_j)$
- Since $\varrho = q_I q_1 q_2 \ldots$ satisfies $\mathcal{M}$'s acceptance condition, Player 1 loses even though he played according to $s$
- $s$ cannot be a winning tree for $t$: we get a contradiction $\Longrightarrow L(s,t) \cap L(\mathcal{M}) = \emptyset$

# Complementation theorem: Proof

How to construct $\mathcal{B}$?

- We got an automaton $\mathcal{M}$ that accepts all sequences over $\Sigma'$ which satisfy $\mathcal{A}$'s acceptance condition
- But we need in a tree automaton $\mathcal{B}$ which recognizes $T(\mathcal{B}) = T_\Sigma \setminus T(\mathcal{A})$
- So to construct $\mathcal{B}$, we first generate a word automaton $\mathcal{S}$ s.t. $L(\mathcal{S}) = \Sigma' \setminus L(\mathcal{M})$
- We can apply Safra's determinization construction [3] to achieve this (details omitted)

# Complementation theorem: Proof

The last thing to show is that indeed $T(\mathcal{B}) = T_\Sigma \setminus T(\mathcal{A})$:

- Assume $t \in T(\mathcal{B})$, i.e. there exists an accepting run $\varrho$ of $\mathcal{B}$ on $t$
- Then for each path $\pi = \pi_1 \pi_2 \cdots \in \{0,1\}^\omega$ the corresponding state sequence satisfies the acceptance condition for the automaton $\mathcal{S}$ we constructed earlier
- So all words $u \in L(s,t)$ are accepted by $\mathcal{S}$
- Since $L(\mathcal{S}) = \Sigma' \setminus L(\mathcal{M})$, $L(s,t) \cap L(\mathcal{M}) = \emptyset$
- Due to Lemma 3, $s$ is a winning tree for Player 1, and $\mathcal{A}$ does not accept $t$

# Complementation theorem: Proof

- Now let $t \notin T(\mathcal{A})$
- Then there exists a winning tree $s$ for Player 1 s.t. $L(s,t) \cap L(\mathcal{M}) = \emptyset$
- It follows $L(s,t) \subseteq \mathcal{S}$, i.e. for each path $\pi = \pi_1\pi_2\cdots \in \{0,1\}^\omega$ there exists a run on the $\omega$-word $u = (s(\varepsilon), t(\varepsilon), \pi_1)(s(\pi_1), t(\pi_1), \pi_2)\cdots \in L(s,t)$ that satisfies the acceptance condition for $\mathcal{S}$
- Hence by construction of $\mathcal{B}$ there exists an accepting run $\varrho$ of $\mathcal{B}$ on $t$, so $t \in T(\mathcal{B})$

# References

[1] Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata Logics, and Infinite Games: A Guide to Current Research*. Springer-Verlag New York, Inc., New York, NY, USA, 2002.

[2] Michael O. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969.

[3] Shmuel Safra. On the complexity of $\omega$-automata. In *Foundations of Computer Science, 1988., 29th Annual Symposium on*, pages 319–327. IEEE, 1988.

[4] Wieslaw Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science*, 200(1–2):135 – 183, 1998.