Non Deterministic Tree Automata
Chapter 8 in the book, by Frank Nießner

Presentation by Rotem Arnon

Games, logic and Automata Seminar 2011
Outline

1. Introduction
   - Motivation
   - Basic Definitions

2. Finite-State Tree Automata
   - Definitions
   - Muller Tree Automaton
   - Parity Tree Automaton
   - Buchi, Rabin and Streett

3. The Complementation Problem for Automata on Infinite Trees
   - Game theoretical approach
   - Complementation proof
1 Introduction
   • Motivation
   • Basic Definitions

2 Finite-State Tree Automata
   • Definitions
   • Muller Tree Automaton
   • Parity Tree Automaton
   • Buchi, Rabin and Streett

3 The Complementation Problem for Automata on Infinite Trees
   • Game theoretical approach
   • Complementation proof
Finite-state Automata for Words

- All the automaton models we have seen so far consume infinite sequences of alphabet symbols.
- We therefore call these automata **word automata**.
Today we define finite-state automata which process infinite trees.

We will call these automata tree automata.
Motivation
Why do we need tree automata?

- Tree automata are more suitable than words when non-determinism needs to be modeled.
Motivation
Why do we need tree automata?

- Tree automata are more suitable than words when non-determinism needs to be modeled.
- Close connections between tree automata and logical theories:
  - For example, decidability of monadic second-order logic using tree automata
1 Introduction
   - Motivation
   - Basic Definitions

2 Finite-State Tree Automata
   - Definitions
   - Muller Tree Automaton
   - Parity Tree Automaton
   - Buchi, Rabin and Streett

3 The Complementation Problem for Automata on Infinite Trees
   - Game theoretical approach
   - Complementation proof
Infinite Binary Tree

Definition

The infinite binary tree is the set $T_\omega = \{0, 1\}^*$ of all finite words on $\{0, 1\}$. The elements $u \in T_\omega$ are the nodes of $T_\omega$ where $\epsilon$ is the root and $u0, u1$ are the immediate left and right successors of node $u$. 

![Diagram of the infinite binary tree](image)
Definition

Let $u, v \in T^\omega$, then $v$ is a **successor** of $u$, denoted by $u < v$, if there exists $w \in T^\omega$ such that $v = uw$. 

\[ T^\omega : \epsilon \]

\[ 0 \quad u \]

\[ 00 \quad 01 \quad v \]

\[ 10 \quad 11 \]

\[ T^\omega : \epsilon \quad u \]

\[ 0 \quad 1 \]

\[ 00 \quad 01 \quad v \]

\[ 10 \quad 11 \]
Infinite Binary Tree

Definition
A path of the binary tree $T_\omega$ is an $\omega$-word $\pi \in \{0, 1\}^\omega$. The set $Pre_<(\pi) \subset \{0, 1\}^*$ of all prefixes of path $\pi$ describes the set of nodes which occur in $\pi$.

Example
The leftmost path in the tree is $\pi = 0^\omega$. All the prefixes of this path are $Pre_<(\pi) = \{\epsilon, 0, 00, 000, 0000, \ldots\} = \{0\}^*$.
Labeling of a Tree

- We would like to consider trees where the nodes are labeled with a symbol of an alphabet $\Sigma$. 
Labeling of a Tree

- We would like to consider trees where the nodes are labeled with a symbol of an alphabet $\Sigma$.

- A mapping $t : T^\omega \rightarrow \Sigma$ labels trees with symbols of $\Sigma$. 
Labeling of a Tree

- We would like to consider trees where the nodes are labeled with a symbol of an alphabet $\Sigma$.

- A mapping $t : T^\omega \rightarrow \Sigma$ labels trees with symbols of $\Sigma$.

- $t \in T^\omega_\Sigma$ denotes some infinite tree with labels form $\Sigma$. 
Labeling of a Tree

- We would like to consider trees where the nodes are labeled with a symbol of an alphabet $\Sigma$.

- A mapping $t : T^\omega \rightarrow \Sigma$ labels trees with symbols of $\Sigma$.

- $t \in T^\omega_\Sigma$ denotes some infinite tree with labels form $\Sigma$.

- $t|_{\pi} : Pre_<(\pi) \rightarrow \Sigma$ denotes the labeling map for a specific path $\pi$. 
Labeling of a Tree

- We would like to consider trees where the nodes are labeled with a symbol of an alphabet $\Sigma$.
- A mapping $t : T^\omega \rightarrow \Sigma$ labels trees with symbols of $\Sigma$.
- $t \in T^\omega_\Sigma$ denotes some infinite tree with labels form $\Sigma$.
- $t|\pi : Pre_<(\pi) \rightarrow \Sigma$ denotes the labeling map for a specific path $\pi$.

**Example**

Let $\Sigma = \{a, b\}$, $t(\epsilon) = a$, and for all $w \in \{0, 1\}^*$, $t(w0) = a$ and $t(w1) = b$. 
Labeling of a Tree

Example
Let $\Sigma = \{a, b\}$, $t(\epsilon) = a$, and for all $w \in \{0, 1\}^*$, $t(w0) = a$ and $t(w1) = b$. 

```
\[ T^\omega : \]
\[
\begin{array}{ccc}
    & \epsilon & \\
   0 & 1 \\
00 & 01 & 10 & 11
\end{array}
\]

\[ t : \]
\[
\begin{array}{ccc}
    & a & b \\
   a & b & a & b
\end{array}
\]
```
1 Introduction
   • Motivation
   • Basic Definitions

2 Finite-State Tree Automata
   • Definitions
     • Muller Tree Automaton
     • Parity Tree Automaton
     • Buchi, Rabin and Streett

3 The Complementation Problem for Automata on Infinite Trees
   • Game theoretical approach
   • Complementation proof
As we said, by now the automata consumed (infinite) words.
As we said, by now the automata consumed (infinite) words.

If we want the automata to accept trees, it needs a new way to read the input.

We will need to modify our definitions.
Each position in a binary tree has two successors (rather than one successor as in infinite words).
Transition Function

- Each position in a binary tree has two successors (rather than one successor as in infinite words).

- It is natural to define for a state out of a set $Q$ and an input symbol from $\Sigma$ two successor states in the transition relation, i.e.,

**Definition**

The transition function $\Delta$ is defined as $\Delta \subseteq (Q \times \Sigma) \times (Q \times Q)$. Alternatively, $\Delta : Q \times \Sigma \rightarrow Q \times Q$.

- Computations start at the root of an input tree and work through the input on each path in parallel.
A Run

- A transition \((q, a, q_1, q_2)\) allows to pass from state \(q\) at node \(u\) with label \(a\), i.e. \(t(u) = a\), to the states \(q_1, q_2\) at the successor nodes \(u_0, u_1\).
- This procedure yields a \(Q\)-labeled tree which we call the run of an automaton on an input tree.
Acceptance Condition

- A run is successful if all the state sequences along the paths meet an acceptance condition.
Acceptance Condition

- A run is successful if all the state sequences along the paths meet an acceptance condition.

- We can define different acceptance conditions:
  - Buchi
  - Muller
  - Rabin
  - Parity
  - and so on...
Acceptance Condition

- A run is successful if all the state sequences along the paths meet an acceptance condition.

- We can define different acceptance conditions:
  - Buchi
  - Muller
  - Rabin
  - Parity
  - and so on...

- Important remark: this means that for each path in the tree the familiar acceptance condition holds.
Introduction

- Motivation
- Basic Definitions

Finite-State Tree Automata

- Definitions
- Muller Tree Automaton
- Parity Tree Automaton
- Buchi, Rabin and Streett

The Complementation Problem for Automata on Infinite Trees

- Game theoretical approach
- Complementation proof
Muller Tree Automaton

Definition

A Muller tree automaton is a quintuple $A = (Q, \Sigma, \Delta, q_i, F)$ where $Q$ is a finite state set, $\Sigma$ is a finite alphabet, $\Delta \subseteq (Q \times \Sigma) \times (Q \times Q)$ denotes the transition relation, $q_i$ is an initial state and $F \in P(Q)$ is a set of state sets. A run of $A$ on an input tree $t \in T_\Sigma^\omega$ is a tree $\rho \in T_Q^\omega$, satisfying $\rho(\epsilon) = q_i$ and for all $w \in \{0, 1\}^*$: $(\rho(w), t(w), \rho(w0), \rho(w1)) \in \Delta$. It is called successful if for each path $\pi \in \{0, 1\}^\omega$ the Muller acceptance condition is satisfied, that is, if $\inf(\rho|\pi) \in F$. 
Muller Tree Automaton

Example

Build a Muller Tree Automata which accepts a tree from $T^\omega_{\{0,1\}}$ if on every path of the tree there is an infinite number of 1’s.
Muller Tree Automaton

**Example**

Build a Muller Tree Automata which accepts a tree from $T_{\{0,1\}^\omega}$ if on every path of the tree there is an infinite number of 1’s.

- $Q = \{q_0, q_1\}$
- $\Sigma = \{0, 1\}$
- $q_i = q_0$
Muller Tree Automaton

Example

Build a Muller Tree Automata which accepts a tree from $T^{\omega}_{\{0,1\}}$ if on every path of the tree there is an infinite number of 1’s.

- $Q = \{q_0, q_1\}$
- $\Sigma = \{0, 1\}$
- $q_i = q_0$
- $\Delta = \{(q_0, 0, q_0, q_0), (q_1, 0, q_0, q_0), (q_0, 1, q_1, q_1), (q_1, 1, q_1, q_1)\}$
Muller Tree Automaton

Example

Build a Muller Tree Automata which accepts a tree from $T^{\omega}_{\{0,1\}}$ if on every path of the tree there is an infinite number of 1’s.

- $Q = \{q_0, q_1\}$
- $\Sigma = \{0, 1\}$
- $q_i = q_0$

- $\Delta =$
  \{(q_0, 0, q_0, q_0), (q_1, 0, q_0, q_0), (q_0, 1, q_1, q_1), (q_1, 1, q_1, q_1)\}$

- $F = \{\{q_1\}, \{q_0, q_1\}\}$
1 Introduction
   - Motivation
   - Basic Definitions

2 Finite-State Tree Automata
   - Definitions
   - Muller Tree Automaton
   - Parity Tree Automaton
   - Buchi, Rabin and Streett

3 The Complementation Problem for Automata on Infinite Trees
   - Game theoretical approach
   - Complementation proof
Parity Tree Automaton

Definition

A Muller tree automaton is a quintuple $\mathcal{A} = (Q, \Sigma, \Delta, q_i, c)$ where $Q$ is a finite state set, $\Sigma$ is a finite alphabet, $\Delta \subseteq (Q \times \Sigma) \times (Q \times Q)$ denotes the transition relation, $q_i$ is an initial state and $c : Q \rightarrow \{0, 1, \ldots, k\}$ for some $k \in \mathbb{N}$ is a function which assigns an index value out of a finite index set to each state of the automaton. A run of $\mathcal{A}$ on an input tree $t \in T^\omega_{\Sigma}$ is a tree $\rho \in T^\omega_Q$, satisfying $\rho(\epsilon) = q_i$ and for all $w \in \{0, 1\}^* : (\rho(w), t(w), \rho(w0), \rho(w1)) \in \Delta$. It is called successful if for each path $\pi \in \{0, 1\}^\omega$ the parity acceptance condition is satisfied, that is, if $\min\{c(q) \mid q \in \inf(\rho|\pi)\}$ is even.
Parity Tree Automaton

Example

Build a parity Tree Automata which accepts a tree from $T^\omega \{0,1\}$ if on every path of the tree there is an infinite number of 1’s.

- $Q = \{q_0, q_1\}$
- $\Sigma = \{0, 1\}$
- $q_i = q_0$
- $\Delta = \{(q_0, 0, q_0, q_0), (q_1, 0, q_0, q_0), (q_0, 1, q_1, q_1), (q_1, 1, q_1, q_1)\}$
- $c(q_0) = 3, \ c(q_1) = 2$
1 Introduction
   • Motivation
   • Basic Definitions

2 Finite-State Tree Automata
   • Definitions
   • Muller Tree Automaton
   • Parity Tree Automaton
   • Buchi, Rabin and Streett

3 The Complementation Problem for Automata on Infinite Trees
   • Game theoretical approach
   • Complementation proof
Buchi, Rabin & Streett

- Buchi, Rabin and Streett tree automata are defined analogously.
- Muller, parity, Rabin and Streett tree automata all accept the same class of languages.
- Buchi tree automata is weaker than the other automaton models.
Buchi Tree Automata is Weaker

Theorem

There exists a Muller tree automaton recognizable language which is not Buchi tree automaton recognizable.

Proof.

On the board.
We will now show closure under complementation for tree languages acceptable by parity tree automata.
Closure Under Complementation

- We will now show closure under complementation for tree languages acceptable by parity tree automata.

- What about Muller tree automata?
1. Introduction
   - Motivation
   - Basic Definitions

2. Finite-State Tree Automata
   - Definitions
   - Muller Tree Automaton
   - Parity Tree Automaton
   - Buchi, Rabin and Streett

3. The Complementation Problem for Automata on Infinite Trees
   - Game theoretical approach
   - Complementation proof
Game Theoretical Approach

- We use a game-theoretical approach.
- We identify a parity tree automaton $\mathcal{A} = (Q, \Sigma, \Delta, q_i, c)$ and an input tree $t$ with an infinite two-person game $\mathcal{G}_{\mathcal{A},t}$. 
Game Theoretical Approach

- We use a game-theoretical approach.
- We identify a parity tree automaton \( A = (Q, \Sigma, \Delta, q_i, c) \) and an input tree \( t \) with an infinite two-person game \( G_{A,t} \).
- Player 0 is called “automaton”.
- Player 1 is called “pathfinder”.
- The players move alternately.
The Rules of the Game $G_{A,t}$

- Player 0, automaton, starts a game by picking an initial transition $(q_i, \sigma, _, _) \in \Delta$ such that the alphabet symbol of this transition equals that at the root of $t$. 
The Rules of the Game $\mathcal{G}_{A,t}$

- Player 0, automaton, starts a game by picking an initial transition $(q_i, \sigma, _, _) \in \Delta$ such that the alphabet symbol of this transition equals that at the root of $t$.

- Player 1, pathfinder, determines whether to proceed with the left or the right successor.
The Rules of the Game $G_{A,t}$

- Player 0, automaton, starts a game by picking an initial transition $(q_i, \sigma, _, _) \in \Delta$ such that the alphabet symbol of this transition equals that at the root of $t$.

- Player 1, pathfinder, determines whether to proceed with the left or the right successor.

- Player 0, automaton, now picks a transition $(q, \sigma, _, _) \in \Delta$ where the alphabet symbol now must equal the input symbol of the left or right successor node in $t$ and the current transition state has to match the left or right successor state of the previous transition.
The Rules of the Game $G_{A,t}$

The tree for the game

Player 0 chose $(q_i, a, q_1, q_0)$

Player 1 chose to move right

Player 0 chose $(q_0, a, q_1)$
Winning the Game

- The sequence of actions represents a play of the game.
- It induces an infinite sequence of states visited along the path across $t$. 
Winning the Game

- The sequence of actions represents a play of the game.

- It induces an infinite sequence of states visited along the path across \( t \).

- Player 0, automaton, wins if this infinite state sequence satisfies the acceptance condition of \( A \).

- Otherwise Player 1 wins.
Winning the Game

- Player 0 wins if this infinite state sequence satisfies the acceptance condition of $A$. Informally:
  - Player 0 will have a winning strategy if all paths of the corresponding run meet the acceptance condition.
  - This means that $A$ accepts the tree $t$.
- Player 1 will have a winning strategy if there exists a path which violates the acceptance condition for every state sequence chosen by player 0.
  - This means that $A$ rejects the tree $t$. 

Presentation by Rotem Arnon
Winning the Game

- Player 0 wins if this infinite state sequence satisfies the acceptance condition of $A$.

Informally:

- Player 0 will have a winning strategy if all paths of the corresponding run meet the acceptance condition.
Winning the Game

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

Informally:

- Player 0 will have a winning strategy if all paths of the corresponding run meet the acceptance condition.

- This means that $\mathcal{A}$ accepts the tree $t$. 
Winning the Game

- Player 0 wins if this infinite state sequence satisfies the acceptance condition of $A$.

Informally:
- Player 0 will have a winning strategy if all paths of the corresponding run meet the acceptance condition.
- This means that $A$ accepts the tree $t$.
- Player 1 will have a winning strategy if there exists a path which violates the acceptance condition for every state sequence chosen by player 0.
Winning the Game

- Player 0 wins if this infinite state sequence satisfies the acceptance condition of $A$.

Informally:
- Player 0 will have a winning strategy if all paths of the corresponding run meet the acceptance condition.
- This means that $A$ accepts the tree $t$.
- Player 1 will have a winning strategy if there exists a path which violates the acceptance condition for every state sequence chosen by player 0.
- This means that $A$ rejects the tree $t$. 
Game Positions
A formal definition of the game

- A play is an infinite sequence of game positions which alternately belong to player 0 or player 1.
- A game can be considered as a graph which consists of all game positions as vertices.
- Edges between different positions indicate that the succeeding position is reachable from the preceding one by a valid action.
Game Positions
A formal definition of the game

- A play is an infinite sequence of game positions which alternately belong to player 0 or player 1.
- A game can be considered as a graph which consists of all game positions as vertices.
- Edges between different positions indicate that the succeeding position is reachable from the preceding one by a valid valid action.

Definitions

1. The game positions of player 0 are defined by
   \[ V_0 = \{ (w, q) \mid w \in \{0, 1\}^*, q \in Q \} \].
2. The game positions of player 1 are defined by
   \[ V_1 = \{ (w, \delta) \mid w \in \{0, 1\}^*, \delta \in \Delta_{t(w)} \} \], where
   \[ \Delta_{t(w)} = \{ \delta \in \Delta \mid \exists q, q_0, q_1 \in Q. \delta = (q, t(w), q_0, q_1) \} \].
Game Positions
A formal definition of the game

- It is clear why the game positions change alternately from a position of $V_0$ and a position from $V_1$. 

The starting position of a play is $(\epsilon, q_i) \in V_0$.

We need to color the vertices (since it is a parity game):

Definition

The coloring function $C : V_0 \cup V_1 \rightarrow \{0, 1, \ldots, k\}$ is defined as:

- $\forall (w, q) \in V_0$. $C((w, q)) = c(q)$
- $\forall (w, (q, t(w), q_0, q_1)) \in V_1$. $C((w, (q, t(w), q_0, q_1))) = c(q)$,

where $c : Q \rightarrow \{0, 1, \ldots, k\}$ is the coloring function of $A$. The games $G_A, t$ meet exactly the definition of min-parity games.
Introduction
Finite-State Tree Automata

The Complementation Problem for Automata on Infinite Trees

Game theoretical approach
Complementation proof

Game Positions
A formal definition of the game

- It is clear why the game positions change alternately from a position of $V_0$ and a position from $V_1$.
- The starting position of a play is $(\epsilon, q_i) \in V_0$. 

Presentation by Rotem Arnon
Non Deterministic Tree Automata
Game Positions
A formal definition of the game

- It is clear why the game positions change alternately from a position of $V_0$ and a position from $V_1$.
- The starting position of a play is $(\epsilon, q_i) \in V_0$.
- We need to color the vertices (since it is a parity game):
Game Positions
A formal definition of the game

- It is clear why the game positions change alternately from a position of $V_0$ and a position from $V_1$.
- The starting position of a play is $(\epsilon, q_i) \in V_0$.
- We need to color the vertices (since it is a parity game):

**Definition**

The **coloring function** $C : V_0 \cup V_1 \rightarrow \{0, 1, \ldots, k\}$ is defined as:

$\forall (w, q) \in V_0. C((w, q)) = c(q)$ and

$\forall (w, (q, t(w), q_0, q_1)) \in V_1. C((w, (q, t(w), q_0, q_1))) = c(q)$,

where $c : Q \rightarrow \{0, 1, \ldots, k\}$ is the coloring function of $A$.

- The games $G_{A, t}$ meet exactly the definition of min-parity games.
The notions of a strategy, a memoryless strategy and a winning strategy are all the same as we know.
The notions of a strategy, a memoryless strategy and a winning strategy are all the same as we know.

A winning strategy of a game $G_{A,t}$ and a successful run $\rho \in T^\omega_Q$ of the corresponding automaton $A = (Q, \Sigma, \Delta, q_i, c)$ are closely related.
Winning Strategy
A formal definition of the game

- The notions of a strategy, a memoryless strategy and a winning strategy are all the same as we know.

- A winning strategy of a game $G_{A,t}$ and a successful run $\rho \in T^\omega_Q$ of the corresponding automaton $A = (Q, \Sigma, \Delta, q_i, c)$ are closely related.

**Theorem**

A tree automaton $A$ accepts an input tree $t$ if and only if there is a winning strategy for player 0 from position $(\epsilon, q_i) \in V_0$ in the game $G_{A,t}$.
Proof - 1’st Direction

Theorem

A tree automaton \( A \) accepts an input tree \( t \) \( \Rightarrow \) there is a winning strategy for player 0 from position \( (\epsilon, q_i) \in V_0 \) in the game \( G_{A,t} \).

- \( A \) accepts the input tree \( t \) \( \Rightarrow \) there exists an accepted run \( \rho \).
Proof - 1’st Direction

Theorem

A tree automaton $A$ accepts an input tree $t \Rightarrow$ there is a winning strategy for player 0 from position $(\epsilon, q_i) \in V_0$ in the game $G_{A,t}$.

- $A$ accepts the input tree $t \Rightarrow$ there exists an accepted run $\rho$.
- The run $\rho$ keeps track of all transitions that have to be chosen in order to accept the input tree $t$. 
Proof - 1’st Direction

Theorem

A tree automaton $A$ accepts an input tree $t \Rightarrow$ there is a winning strategy for player 0 from position $(\epsilon, q_i) \in V_0$ in the game $G_{A,t}$.

- $A$ accepts the input tree $t \Rightarrow$ there exists an accepted run $\rho$.

- The run $\rho$ keeps track of all transitions that have to be chosen in order to accept the input tree $t$.

- For any of the nodes $(w, q) \in V_0$, where $(w_0, q_0)$ and $(w_1, q_1)$ are the immediate successors, we can derive the corresponding transition $\delta = (q, t(w), q_0, q_1) \in \Delta$. 
Proof - 1’st Direction

Theorem

A tree automaton $A$ accepts an input tree $t \Rightarrow$ there is a winning strategy for player 0 from position $(\epsilon, q_i) \in V_0$ in the game $G_{A,t}$.

- Player 0 can always choose the right transition, independently of player 1’s decisions.
**Theorem**

A tree automaton $A$ accepts an input tree $t$ $\Rightarrow$ there is a winning strategy for player 0 from position $(\epsilon, q_i) \in V_0$ in the game $G_{A,t}$.

- Player 0 can always choose the right transition, independently of player 1’s decisions.
- This means that if player 0 will choose his movements according to $\rho$ he will always win (since $\rho$ is an accepted run).
Proof - 2’nd Direction

**Theorem**

There is a winning strategy for player 0 from position $(\varepsilon, q_i) \in V_0$ in the game $G_{A,t} \Rightarrow A$ tree automaton $A$ accepts an input tree $t$.

- We can use the winning strategy $f_0$ for player 0 in $G_{A,t}$ to construct a successful run of $A$ on $t$. 
Proof - 2’nd Direction

Theorem

There is a winning strategy for player 0 from position \((\epsilon, q_i) \in V_0\) in the game \(G_{A,t}\) ⇒ A tree automaton \(A\) accepts an input tree \(t\).

- We can use the winning strategy \(f_0\) for player 0 in \(G_{A,t}\) to construct a successful run of \(A\) on \(t\).

- For each game position \((w, q) \in V_0\), \(f_0\) determines the correct transition \(\delta = (q, t(w), q_0, q_1) \in \Delta\).
Proof - 2’nd Direction

**Theorem**

There is a winning strategy for player 0 from position 
\((\epsilon, q_i) \in V_0\) in the game \(G_{A,t}\) ⇒ A tree automaton \(A\) accepts an input tree \(t\).

- We can use the winning strategy \(f_0\) for player 0 in \(G_{A,t}\) to construct a successful run of \(A\) on \(t\).

- For each game position \((w, q) \in V_0\), \(f_0\) determines the correct transition \(\delta = (q, t(w), q_0, q_1) \in \Delta\).

- Player 0 must be prepared to proceed form both \((w0, q_0)\) and \((w1, q_1)\) since he can’t predict player 1’s decision.
Player 0 must be prepared to proceed from both \((w_0, q_0)\) and \((w_1, q_1)\) since he can’t predict player 1’s decision.
Proof - 2’nd Direction

- Player 0 must be prepared to proceed form both \((w_0, q_0)\) and \((w_1, q_1)\) since he can’t predict player 1’s decision.

- Hence we label \(w\) by \(q\), \(w_0\) by \(q_0\) and \(w_1\) by \(q_1\). Proceeding this way we obtain the entire run \(\rho\).
Proof - 2’nd Direction

- Player 0 must be prepared to proceed from both \((w_0, q_0)\) and \((w_1, q_1)\) since he can’t predict player 1’s decision.

- Hence we label \(w\) by \(q\), \(w_0\) by \(q_0\) and \(w_1\) by \(q_1\). Proceeding this way we obtain the entire run \(\rho\).

- \(\rho\) is successful since it is determined by a winning strategy \(f_0\).
Winning Strategy

**Theorem**

A tree automaton $A$ accepts an input tree $t$ if and only if there is a winning strategy for player 0 from position $(\epsilon, q_i) \in V_0$ in the game $G_{A,t}$. 
Winning Strategy

**Theorem**

A tree automaton $A$ accepts an input tree $t$ if and only if there is a winning strategy for player 0 from position $(\epsilon, q_i) \in V_0$ in the game $G_{A,t}$.

**Fact**

Parity games are determined and memoryless winning strategies suffice to win a parity game.
Winning Strategy

**Theorem**

A tree automaton \( A \) accepts an input tree \( t \) if and only if there is a winning strategy for player 0 from position \( (\epsilon, q_i) \in V_0 \) in the game \( G_{A,t} \).

**Fact**

Parity games are determined and memoryless winning strategies suffice to win a parity game.

**Corollary**

From any game position in \( G_{A,t} \), either player 0 or player 1 has a memoryless winning strategy.
1 Introduction
   - Motivation
   - Basic Definitions

2 Finite-State Tree Automata
   - Definitions
   - Muller Tree Automaton
   - Parity Tree Automaton
   - Buchi, Rabin and Streett

3 The Complementation Problem for Automata on Infinite Trees
   - Game theoretical approach
   - Complementation proof
Given a parity tree automaton $A$, we have to specify a tree automaton $B$ that accepts all input trees rejected by $A$. 
The Complementation of Finite Tree Automata Languages

- Given a parity tree automaton $A$, we have to specify a tree automaton $B$ that accepts all input trees rejected by $A$.

- Rejecting means that there is no winning strategy for player 0 from position $(\epsilon, q_i)$ in the game $G_{A,t}$.
Given a parity tree automaton $A$, we have to specify a tree automaton $B$ that accepts all input trees rejected by $A$.

Rejecting means that there is no winning strategy for player 0 from position $(\epsilon, q_i)$ in the game $G_{A,t}$.

This guarantees the existence of a memoryless winning strategy starting at $(\epsilon, q_i)$ for player 1.

We will construct an automaton that checks exactly this.
A memoryless strategy of player 1 is a function $f_1 : \{0, 1\}^* \times \Delta \rightarrow \{0, 1\}$ determining a direction 0 (left successor) or 1 (right successor).
Memoryless Strategy of Player 1

- A memoryless strategy of player 1 is a function $f_1 : \{0, 1\}^* \times \Delta \rightarrow \{0, 1\}$ determining a direction 0 (left successor) or 1 (right successor).

- There is a natural isomorphism between such functions and functions $f_1 : \{0, 1\}^* \rightarrow (\Delta \rightarrow \{0, 1\})$. 
Memoryless Strategy of Player 1

- A memoryless strategy of player 1 is a function $f_1 : \{0, 1\}^* \times \Delta \rightarrow \{0, 1\}$ determining a direction 0 (left successor) or 1 (right successor).

- There is a natural isomorphism between such functions and functions $f_1 : \{0, 1\}^* \rightarrow (\Delta \rightarrow \{0, 1\})$.

- $f_1 : \{0, 1\}^* \rightarrow (\Delta \rightarrow \{0, 1\})$ is a tree (with functions as labels).
Memoryless Strategy of Player 1

- A memoryless strategy of player 1 is a function $f_1 : \{0, 1\}^* \times \Delta \to \{0, 1\}$ determining a direction 0 (left successor) or 1 (right successor).

- There is a natural isomorphism between such functions and functions $f_1 : \{0, 1\}^* \to (\Delta \to \{0, 1\})$.

- $f_1 : \{0, 1\}^* \to (\Delta \to \{0, 1\})$ is a tree (with functions as labels).

- We call such trees strategy trees.

- If the corresponding strategy is winning for player 1 in the game $G_{A,t}$, we say it is a winning tree for $t$. 
From these definitions and all of the above it is clear that:

**Fact**

Let $A$ be a parity tree automaton and $t$ be an input tree. There exists a *winning tree* $s$ for player 1 if and only if $A$ does not accept $t$. 
Building the Automaton $\mathcal{B}$

In order to build the tree automaton $\mathcal{B}$ which will accept $\overline{L(\mathcal{A})}$ we will use a $\omega$-words automaton $\mathcal{M}$. 

Informally: $\mathcal{M}$ will get a path in the tree, with its corresponding labels of $t$ and corresponding strategies for player 1 (labels of $s$). $\mathcal{M}$ will then decide if this path, using the strategy for player 1 defined by $s$, will be accepted by $\mathcal{A}$. If yes, it accepts. $\mathcal{M}$ will need to check all the possible strategies for player 0.

Therefore $L(\mathcal{M})$ will be all the paths + strategies of player 1 which are good for player 0.
Building the Automaton $B$

- In order to build the tree automaton $B$ which will accept $L(A)$ we will use a $\omega$-words automaton $M$.

Informally:
- $M$ will get a path in the tree, with its corresponding labels of $t$ and corresponding strategies for player 1 (labels of $s$).
Building the Automaton $\mathcal{B}$

- In order to build the tree automaton $\mathcal{B}$ which will accept $\overline{L(\mathcal{A})}$ we will use a $\omega$-words automaton $\mathcal{M}$.

Informally:

- $\mathcal{M}$ will get a path in the tree, with its corresponding labels of $t$ and corresponding strategies for player 1 (labels of $s$).

- $\mathcal{M}$ will then decide if this path, using the strategy for player 1 defined by $s$, will be accepted by $\mathcal{A}$. If yes it accepts.
Building the Automaton $\mathcal{B}$

- In order to build the tree automaton $\mathcal{B}$ which will accept $L(\mathcal{A})$, we will use a $\omega$-words automaton $\mathcal{M}$.

Informally:

- $\mathcal{M}$ will get a path in the tree, with its corresponding labels of $t$ and corresponding strategies for player 1 (labels of $s$).

- $\mathcal{M}$ will then decide if this path, using the strategy for player 1 defined by $s$, will be accepted by $\mathcal{A}$. If yes it accepts.

- $\mathcal{M}$ will need to check all the possible strategies for player 0.
Building the Automaton $B$

- In order to build the tree automaton $B$ which will accept $\overline{L(A)}$ we will use a $\omega$-words automaton $M$.

Informally:

- $M$ will get a path in the tree, with its corresponding labels of $t$ and corresponding strategies for player 1 (labels of $s$).

- $M$ will then decide if this path, using the strategy for player 1 defined by $s$, will be accepted by $A$. If yes it accepts.

- $M$ will need to check all the possible strategies for player 0.

- Therefore $L(M)$ will be all the paths + strategies of player 1 which are good for player 0.
Building the Automaton $\mathcal{M}$

**Definition**

Let $L(s, t)$ be the language of all of $\omega$-words of the type $(s(\epsilon), t(\epsilon), \pi_1)(s(\pi_1), t(\pi_1), \pi_2)\ldots$. I.e., every path $\pi$ defines a different word in $L(s, t)$.

**Example**

Consider a path $\pi = 0110\ldots$ through the tree $t$. An $\omega$-word $u \in L(s, t)$ determined by $\pi$ could look like

$$(f_\epsilon, t(\epsilon), 0)(f_0, t(0), 1)(f_{01}, t(01), 1)(f_{011}, t(011), 0)\ldots$$
Building the Automaton $\mathcal{M}$

**Definition**

$\mathcal{M} = (Q, \Sigma', \Lambda, q_i, c)$ where

$\Sigma' = \{(f, a, i) | f : \Delta \rightarrow \{0, 1\}, a \in \Sigma, i \in \{0, 1\}\}$, $\mathcal{A}$ and $\mathcal{M}$ have the same acceptance condition.

$\Lambda$ is defined as follows: for $(f, a, i) \in \Sigma'$, $f : \Delta_a \rightarrow \{0, 1\}$, and $\delta = (q, a, q_0, q_1) \in \Delta_a$ such that $f(\delta) = i$, $\mathcal{M}$ has a transition $(q, (f, a, i), q_i)$.

$\mathcal{M}$ checks each possible move of player 0 if the outcome is winning for player 0. This is done nondeterministically.
Building the Automaton $M$

Informally:

- $M$ first checks if the given path in $t$ is consistent with player 1’s strategy according to $s$.
- Because $M$ uses the same acceptance condition as $A$ it will accept if the run on this path is consistent with $s$ and will be accepted by $A$. 
Building the Automaton $M$

Informally:

- $M$ first checks if the given path in $t$ is consistent with player 1’s strategy according to $s$.

- Because $M$ uses the same acceptance condition as $A$ it will accept if the run on this path is consistent with $s$ and will be accepted by $A$.

- If $M$ won’t accept an $\omega$-word $u$ it means that player 1 will go through the path $\pi$ which defined $u$, and win this way (since $M$ checked all the options for player 0 and non of them worked).
Theorem

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

Informally:
All the possible paths that player 1 can choose, while using strategy $s$, define the words in $L(s, t)$. $L(M)$ holds all the paths which are good for player 0 and consistent with player 1's strategy. If $L(s, t) \cap L(M) = \emptyset$ then all the paths in $t$ which are consistent with $s$ will make player 1 win, i.e., $s$ is a winning strategy for player 1.
Using $\mathcal{M}$

**Theorem**

The tree $s$ is a winning tree for $t$ if and only if

$L(s, t) \cap L(\mathcal{M}) = \emptyset$.

Informally:

- All the possible paths that player 1 can choose, while using strategy $s$, define the words in $L(s, t)$. 

Using $\mathcal{M}$

**Theorem**

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

Informally:

- All the possible paths that player 1 can choose, while using strategy $s$, define the words in $L(s, t)$.

- $L(\mathcal{M})$ holds all the paths which are good for player 0 and consistent with player 1’s strategy.
Using $\mathcal{M}$

**Theorem**

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

Informally:

- All the possible paths that player 1 can choose, while using strategy $s$, define the words in $L(s, t)$.

- $L(\mathcal{M})$ holds all the paths which are good for player 0 and consistent with player 1’s strategy.

- If $L(s, t) \cap L(\mathcal{M}) = \emptyset$ then all the paths in $t$ which are consistent with $s$ will make player 1 win, i.e., $s$ is a winning strategy for player 1.
Theorem

If the tree $s$ is a winning tree for $t$ then $L(s, t) \cap L(M) = \emptyset$. 

Presentation by Rotem Arnon
Proof - 1’st Direction

**Theorem**

*If the tree $s$ is a winning tree for $t$ then $L(s, t) \cap L(M) = \emptyset$.*

- Let $s$ be a winning tree.
- Assume in contradiction that $L(s, t) \cap L(M) \neq \emptyset$, i.e., there exists a path $\pi = \pi_1 \pi_2 \ldots \in \{0, 1\}^\omega$ such that the corresponding $\omega$-word $u = (s(\epsilon), t(\epsilon), \pi_1) (s(\pi_1), t(\pi_1), \pi_2) \ldots \in L(M)$.
Introduction
Finite-State Tree Automata

The Complementation Problem for Automata on Infinite Trees

Game theoretical approach
Complementation proof

Proof - 1’st Direction

Theorem

If the tree $s$ is a winning tree for $t$ then $L(s, t) \cap L(M) = \emptyset$.

- Let $s$ be a winning tree.
- Assume in contradiction that $L(s, t) \cap L(M) \neq \emptyset$, i.e., there exists a path $\pi = \pi_1 \pi_2 \ldots \in \{0, 1\}^\omega$ such that the corresponding $\omega$-word $u = (s(\epsilon), t(\epsilon), \pi_1) (s(\pi_1), t(\pi_1), \pi_2) \ldots \in L(M)$.
- So there is a successful run $\rho = q_i q_1 q_2 \ldots$ of $M$ on $u$. 
Proof - 1’st Direction

**Theorem**

*If the tree $s$ is a winning tree for $t$ then $L(s, t) \cap L(M) = \emptyset$.***

- Let $s$ be a winning tree.
- Assume in contradiction that $L(s, t) \cap L(M) \neq \emptyset$, i.e., there exists a path $\pi = \pi_1 \pi_2 \ldots \in \{0, 1\}^\omega$ such that the corresponding $\omega$-word $u = (s(\epsilon), t(\epsilon), \pi_1) (s(\pi_1), t(\pi_1), \pi_2) \ldots \in L(M)$.
- So there is a successful run $\rho = q_i q_1 q_2 \ldots$ of $M$ on $u$.
- Therefore for each transition $(q_j, (s(\pi_1 \ldots \pi_j), t(\pi_1 \ldots \pi_j), \pi_{j+1}), q_{j+1})$ that occurs in $\rho$ there is an appropriate transition $\delta_j = (q_j, t(\pi_1 \ldots \pi_j), q_0, q_1)$ of $A$ such that $s(\pi_1 \ldots \pi_j) = f_{\pi_1 \ldots \pi_j}$ where $f_{\pi_1 \ldots \pi_j}(\delta_j) = \pi_{j+1}$. If $\pi_{j+1} = 0$ then $q_{j+1} = q_0$ and if $\pi_{j+1} = q$ then $q_{j+1} = q_1$. 
Proof - 1’st Direction

Let these transitions $\delta_j$ be player 0’s choices in a play of $G_{A,t}$ where player 1 reacts by choosing $s(\pi_1...\pi_j)$. 

Hence player 1 loses even though he played according to $s$. $s$ is not a winning tree.
Proof - 1’st Direction

- Let these transitions $\delta_j$ be player 0’s choices in a play of $\mathcal{G}_{A,t}$ where player 1 reacts by choosing $s(\pi_1...\pi_j)$.

- The sequence of states visited along this play satisfies $M$’s acceptance condition.
Proof - 1’st Direction

- Let these transitions $\delta_j$ be player 0’s choices in a play of $G_{A,t}$ where player 1 reacts by choosing $s(\pi_1...\pi_j)$.

- The sequence of states visited along this play satisfies $M$’s acceptance condition.

- Hence player 1 loses even though he played according to $s$. 
Proof - 1st Direction

- Let these transitions $\delta_j$ be player 0’s choices in a play of $G_{A,t}$ where player 1 reacts by choosing $s(\pi_1...\pi_j)$.

- The sequence of states visited along this play satisfies $M$’s acceptance condition.

- Hence player 1 loses even though he played according to $s$.

- $s$ is not a winning tree.
Proof - 2’nd Direction

Theorem

If $L(s, t) \cap L(M) = \emptyset$ then the tree $s$ is a winning tree for $t$. 
Proof - 2’nd Direction

Theorem

If \( L(s, t) \cap L(M) = \emptyset \) then the tree \( s \) is a winning tree for \( t \).

- Let \( L(s, t) \cap L(M) = \emptyset \).
Proof - 2’nd Direction

Theorem

If $L(s, t) \cap L(M) = \emptyset$ then the tree $s$ is a winning tree for $t$.

- Let $L(s, t) \cap L(M) = \emptyset$.
- We consider any play of the game $G_{A,t}$ and assume $\delta_j = (q_j, t(\pi_1...\pi_j), q_0, q_1) \in \Delta$ to be player 0’s choice when $\pi_1...\pi_j$ is the current node.
Proof - 2’nd Direction

**Theorem**

*If* $L(s, t) \cap L(\mathcal{M}) = \emptyset$ *then the tree* $s$ *is a winning tree for* $t$.

- Let $L(s, t) \cap L(\mathcal{M}) = \emptyset$.
- We consider any play of the game $\mathcal{G}_{A,t}$ and assume $\delta_j = (q_j, t(\pi_1...\pi_j), q_0, q_1) \in \Delta$ to be player 0’s choice when $\pi_1...\pi_j$ is the current node.
- Player 1 plays according to $s$. 
Introduction

Finite-State Tree Automata

The Complementation Problem for Automata on Infinite Trees

Game theoretical approach

Complementation proof

Proof - 2’nd Direction

Theorem

If \( L(s, t) \cap L(M) = \emptyset \) then the tree \( s \) is a winning tree for \( t \).

- Let \( L(s, t) \cap L(M) = \emptyset \).

- We consider any play of the game \( G_{A,t} \) and assume \( \delta_j = (q_j, t(\pi_1...\pi_j), q_0, q_1) \in \Delta \) to be player 0’s choice when \( \pi_1...\pi_j \) is the current node.

- Player 1 plays according to \( s \).

- The successor state is determined by \( s(\pi_1...\pi_j) \), i.e., \( q_{j+1} \in \{q_0, q_1\} \).
Theorem

If \( L(s, t) \cap L(M) = \emptyset \) then the tree \( s \) is a winning tree for \( t \).

- Let \( L(s, t) \cap L(M) = \emptyset \).
- We consider any play of the game \( G_{A,t} \) and assume \( \delta_j = (q_j, t(\pi_1 ... \pi_j), q_0, q_1) \in \Delta \) to be player 0’s choice when \( \pi_1 ... \pi_j \) is the current node.
- Player 1 plays according to \( s \).
- The successor state is determined by \( s(\pi_1 ... \pi_j) \), i.e., \( q_{j+1} \in \{q_0, q_1\} \).
- In this way we obtain an infinite sequence \( \rho = q_1 q_2 ... \) of states visited along the play.
Proof - 2’nd Direction

- This sequence is as well the run of $M$ on the corresponding $\omega$-word $u = (s(\epsilon), t(\epsilon), \pi_1) (s(\pi_1), t(\pi_1), \pi_2) \ldots \in L(s, t)$. 

Since $L(s, t) \cap L(M) = \emptyset$, $\rho$ is not accepting. 

The run $\rho$ is a particular path of $A$’s run on $t$ which is determined by player 0’s choices. 

This implies that $A$ cannot accept $t$ by this run. 

These observations hold for any run $t / \in L(A)$. 

$s$ is a winning tree for $t$. 

Presentation by Rotem Arnon
Proof - 2’nd Direction

- This sequence is as well the run of $\mathcal{M}$ on the corresponding $\omega$-word $u = (s(\epsilon), t(\epsilon), \pi_1) (s(\pi_1), t(\pi_1), \pi_2) \ldots \in L(s, t)$.

- Since $L(s, t) \cap L(\mathcal{M}) = \emptyset$, $\rho$ is not accepting.
Proof - 2’nd Direction

- This sequence is as well the run of $M$ on the corresponding $\omega$-word $u = (s(\epsilon), t(\epsilon), \pi_1) (s(\pi_1), t(\pi_1), \pi_2) \ldots \in L(s, t)$.

- Since $L(s, t) \cap L(M) = \emptyset$, $\rho$ is not accepting.

- The run $\rho$ is a particular path of $A$’s run on $t$ which is determined by player 0’s choices.
Proof - 2’nd Direction

- This sequence is as well the run of $\mathcal{M}$ on the corresponding $\omega$-word $u = (s(\epsilon), t(\epsilon), \pi_1) (s(\pi_1), t(\pi_1), \pi_2) \ldots \in L(s, t)$.

- Since $L(s, t) \cap L(\mathcal{M}) = \emptyset$, $\rho$ is not accepting.

- The run $\rho$ is a particular path of $\mathcal{A}$’s run on $t$ which is determined by player 0’s choices.

- This implies that $\mathcal{A}$ cannot accept $t$ by this run.
Proof - 2’nd Direction

- This sequence is as well the run of $\mathcal{M}$ on the corresponding $\omega$-word $u = (s(\epsilon), t(\epsilon), \pi_1) (s(\pi_1), t(\pi_1), \pi_2) \ldots \in L(s, t)$.

- Since $L(s, t) \cap L(\mathcal{M}) = \emptyset$, $\rho$ is not accepting.

- The run $\rho$ is a particular path of $\mathcal{A}$’s run on $t$ which is determined by player 0’s choices.

- This implies that $\mathcal{A}$ cannot accept $t$ by this run.

- These observations hold for any run.
Proof - 2’nd Direction

- This sequence is as well the run of $\mathcal{M}$ on the corresponding $\omega$-word $u = (s(\epsilon), t(\epsilon), \pi_1)(s(\pi_1), t(\pi_1), \pi_2) \ldots \in L(s, t)$.

- Since $L(s, t) \cap L(\mathcal{M}) = \emptyset$, $\rho$ is not accepting.

- The run $\rho$ is a particular path of $\mathcal{A}$’s run on $t$ which is determined by player 0’s choices.

- This implies that $\mathcal{A}$ cannot accept $t$ by this run.

- These observations hold for any run.

- $t \notin L(\mathcal{A})$.

- $s$ is a winning tree for $t$. 

Building $\mathcal{B}$ from $\mathcal{M}$

- The word automaton $\mathcal{M}$ accepts all sequences over $\Sigma'$ which satisfy $\mathcal{A}$’s acceptance condition.
Building $B$ from $M$

- The word automaton $M$ accepts all sequences over $\Sigma'$ which satisfy $A$'s acceptance condition.
- We are interested in a tree automaton $B$ which recognizes $T(B) = T_\omega^\omega \setminus T(A)$. 

Presentation by Rotem Arnon
Building $B$ from $M$

- The word automaton $M$ accepts all sequences over $\Sigma'$ which satisfy $A$’s acceptance condition.

- We are interested in a tree automaton $B$ which recognizes $T(B) = T_\Sigma^\omega \setminus T(A)$.

- We first build a word automaton $S$ such that $L(S) = \Sigma' \setminus L(M)$. 
Building $\mathcal{B}$ from $\mathcal{M}$

- The word automaton $\mathcal{M}$ accepts all sequences over $\Sigma'$ which satisfy $\mathcal{A}$’s acceptance condition.

- We are interested in a tree automaton $\mathcal{B}$ which recognizes $T(\mathcal{B}) = T_\Sigma^\omega \setminus T(\mathcal{A})$.

- We first build a word automaton $\mathcal{S}$ such that $L(\mathcal{S}) = \Sigma' \setminus L(\mathcal{M})$.

- This we can do since it is a word automaton. $\mathcal{S} = (Q', \Sigma', \gamma, q'_i, \Omega)$.
Building $\mathcal{B}$ from $\mathcal{M}$

- The word automaton $\mathcal{M}$ accepts all sequences over $\Sigma'$ which satisfy $\mathcal{A}$'s acceptance condition.

- We are interested in a tree automaton $\mathcal{B}$ which recognizes $T(\mathcal{B}) = T_\Sigma^\omega \setminus T(\mathcal{A})$.

- We first build a word automaton $\mathcal{S}$ such that $L(\mathcal{S}) = \Sigma' \setminus L(\mathcal{M})$.

- This we can do since it is a word automaton. $\mathcal{S} = (Q', \Sigma', \gamma, q'_i, \Omega)$

- We then use $\mathcal{S}$ in order to build $\mathcal{B}$. $\mathcal{B} = (Q', \Sigma, \Delta', q_i, \Omega)$
Building $\mathcal{B}$ from $\mathcal{S}$

- $\mathcal{B}$ will run $\mathcal{S}$ in parallel along each path of an input tree.
Building $B$ from $S$

- $B$ will run $S$ in parallel along each path of an input tree.

- The transition relation of $B$ is defined by: $(q, a, q_0, q_1) \in \Delta'$ if and only if there exist transitions $\delta(q, (f, a, 0)) = q_0$ and $\delta(q, (f, a, 1)) = q_1$ in $S$ where $f : \Delta_a \rightarrow \{0, 1\}$. 

Theorem

The class of languages recognized by finite-state tree automata is closed under complementation. It remains to be shown that indeed $T(B) = T(\omega \Sigma \setminus T(A))$. 

Presentation by Rotem Arnon
Building $\mathcal{B}$ from $\mathcal{S}$

- $\mathcal{B}$ will run $\mathcal{S}$ in parallel along each path of an input tree.

- The transition relation of $\mathcal{B}$ is defined by: $(q, a, q_0, q_1) \in \Delta'$ if and only if there exist transitions $\delta(q, (f, a, 0)) = q_0$ and $\delta(q, (f, a, 1)) = q_1$ in $\mathcal{S}$ where $f : \Delta_a \to \{0, 1\}$.

Theorem

The class of languages recognized by finite-state tree automata is closed under complementation.
Building $B$ from $S$

- $B$ will run $S$ in parallel along each path of an input tree.

- The transition relation of $B$ is defined by: $(q, a, q_0, q_1) \in \Delta'$ if and only if there exist transitions $\delta(q, (f, a, 0)) = q_0$ and $\delta(q, (f, a, 1)) = q_1$ in $S$ where $f : \Delta_a \rightarrow \{0, 1\}$.

**Theorem**

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

- It remains to be shown that indeed $T(B) = T_{\Sigma}^\omega \setminus T(A)$. 
Proof - 1'\textsuperscript{st} Direction

- Assume $t \in T(\mathcal{B})$, i.e., there exists an accepting run $\rho$ of $\mathcal{B}$ on $t$. 
Proof - 1'st Direction

- Assume \( t \in T(B) \), i.e., there exists an accepting run \( \rho \) of \( B \) on \( t \).

- Hence for each path \( \pi = \pi_1 \pi_2 ... \in \{0, 1\}^\omega \) the corresponding state sequence satisfies \( B \)'s acceptance condition, \( \Omega \) (same as in \( S \)).
Proof - 1’st Direction

- Assume $t \in T(B)$, i.e., there exists an accepting run $\rho$ of $B$ on $t$.

- Hence for each path $\pi = \pi_1 \pi_2 \ldots \in \{0, 1\}^\omega$ the corresponding state sequence satisfies $B$’s acceptance condition, $\Omega$ (same as in $S$).

- For each node $w \in \{0, 1\}^\omega$ there are transitions $
\delta(q, (s(w), t(w), 0)) = q_0$ and $\delta(q, (s(w), t(w), 1)) = q_1$ of $S$ where $s(w) : \Delta_{t(w)} \rightarrow \{0, 1\}$ and the corresponding transition of $B$ is $(q, t(w), q_0, q_1)$. 

Proof - 1'st Direction

- Assume $t \in T(B)$, i.e., there exists an accepting run $\rho$ of $B$ on $t$.

- Hence for each path $\pi = \pi_1 \pi_2 ... \in \{0, 1\}^\omega$ the corresponding state sequence satisfies $B$'s acceptance condition, $\Omega$ (same as in $S$).

- For each node $w \in \{0, 1\}^\omega$ there are transitions $\delta(q, (s(w), t(w), 0)) = q_0$ and $\delta(q, (s(w), t(w), 1)) = q_1$ of $S$ where $s(w) : \Delta_{t(w)} \to \{0, 1\}$ and the corresponding transition of $B$ is $(q, t(w), q_0, q_1)$.

- This implies that all words $u \in L(s, t)$ are accepted by $S$. 
Proof - 1’st Direction

- This implies that all words \( u \in L(s, t) \) are accepted by \( S \), i.e., \( L(s, t) \subseteq L(S) \).
Proof - 1’st Direction

- This implies that all words $u \in L(s, t)$ are accepted by $S$, i.e., $L(s, t) \subseteq L(S)$.

- $L(S) = \Sigma' \setminus L(M) \implies L(s, t) \cap L(M) = \emptyset$
Proof - 1’st Direction

- This implies that all words $u \in L(s, t)$ are accepted by $S$, i.e., $L(s, t) \subseteq L(S)$.

- $L(S) = \Sigma' \setminus L(M) \implies L(s, t) \cap L(M) = \emptyset$

- We proved: The tree $s$ is a winning tree for $t$ if and only if $L(s, t) \cap L(M) = \emptyset$. 
Proof - 1’st Direction

- This implies that all words $u \in L(s, t)$ are accepted by $S$, i.e., $L(s, t) \subseteq L(S)$.

- $L(S) = \Sigma' \setminus L(M) \implies L(s, t) \cap L(M) = \emptyset$

- We proved: The tree $s$ is a winning tree for $t$ if and only if $L(s, t) \cap L(M) = \emptyset$.

- Therefore $s$ is a winning tree for player 1 and $A$ does not accept $t$.

- $t \notin T(A)$
Proof - 2’nd Direction

- Assume \( t \notin T(A) \), i.e., there exists a winning tree \( s \) for player 1.
Proof - 2’nd Direction

• Assume $t \notin T(A)$, i.e., there exists a winning tree $s$ for player 1.

• $L(s, t) \cap L(M) = \emptyset \implies L(s, t) \subseteq L(S)$
Proof - 2’nd Direction

- Assume \( t \not\in T(A) \), i.e., there exists a winning tree \( s \) for player 1.

- \( L(s, t) \cap L(M) = \emptyset \implies L(s, t) \subseteq L(S) \)

- For each path \( \pi = \pi_1 \pi_2 \ldots \in \{0, 1\}^\omega \) there exists a run on the \( \omega \)-word \( u = (s(\epsilon), t(\epsilon), \pi_1) (s(\pi_1), t(\pi_1), \pi_2) \ldots \in L(s, t) \) that satisfies \( \Omega \).
Proof - 2’nd Direction

- Assume \( t \notin T(A) \), i.e., there exists a winning tree \( s \) for player 1.

- \( L(s, t) \cap L(M) = \emptyset \implies L(s, t) \subseteq L(S) \)

- For each path \( \pi = \pi_1 \pi_2 \ldots \in \{0, 1\}^\omega \) there exists a run on the \( \omega \)-word \( u = (s(\epsilon), t(\epsilon), \pi_1)(s(\pi_1), t(\pi_1), \pi_2) \ldots \in L(s, t) \) that satisfies \( \Omega \).

- Hence by construction of \( B \) there exists an accepting run \( \rho \) of \( B \) on \( t \).

- \( t \in T(B) \)
All Together

- From $A$ (tree automaton) we built $M$ (words automaton).
- From $M$ we built $S$ such that $L(S) = \overline{L(M)}$
- From $S$ we built $B$
1 Introduction
   - Motivation
   - Basic Definitions

2 Finite-State Tree Automata
   - Definitions
   - Muller Tree Automaton
   - Parity Tree Automaton
   - Buchi, Rabin and Streett

3 The Complementation Problem for Automata on Infinite Trees
   - Game theoretical approach
   - Complementation proof