Tree Automata
The Model $T_2$

The structure of the infinite binary tree is

$$T_2 = (\{0, 1\}^*, S_0, S_1, \varepsilon)$$

where $S_i$ is the $i$-th successor function:

$$S_0(u) = u0, \quad S_1(u) = u1$$

The theory S2S is set of S2S-sentences which are true in $T_2$.

It is also called the monadic theory of the binary tree,

denoted by $\text{MTh}_2(T_2)$.

Our aim: $\text{MTh}_2(T_2)$ ist decidable.
Problem 1. Let SC² be like SC, except that the functions 2x+1 and 2x+2 are taken as primitives in place of x+1. Is SC² decidable?

This is of some interest, because the functions 2x+1 and 2x+2 can be interpreted as the right-successor functions x1 and x2 on the set of all words on two generators 1 and 2.

Problem 2. Let SC(α) be like SC, except that the domain of individuals is the ordinal α, and the well ordering on α is added as a primitive. Is SC(ω²) decidable?

As outlined in the introduction, Theorem 2 may be interpreted as a method for deciding whether or not a given finite automaton satisfies a given condition in SC.

Problem 3. Is there a solvability algorithm for SC, i.e., is there a method which applies to any formula C(i, u) of SC and decides whether or not there is a finite automata recursion A(i, r, u) which satisfies the condition C (i.e., A(i, r, u) ⊆ C(i, u))?
Example Formulas

Definition of $x \preceq y$ (“node $x$ is prefix of node $y$”):

$\varphi^*_s(x, y)$ with $\varphi_s(z, z') := z_0 = z' \lor z_1 = z'$

$$\forall X ((X(y) \land \forall z (X(z_0) \rightarrow X(z))) \land \forall z (X(z_1) \rightarrow X(z))) \rightarrow X(x)$$

```
Chain(X)  ("X is linearly ordered by $\preceq$"):

\forall x \forall y ((X(x) \land X(y)) \rightarrow (x \preceq y \lor y \preceq x))
```

```
Path(X)  ("X is a path, i.e. a maximal chain"):

Chain(X) \land \neg \exists Y (X \subseteq Y \land X \neq Y \land Chain(Y))
```

```
X \subseteq Y: \forall z (X(z) \rightarrow Y(z))
```

```
X = Y: \forall z (X(z) \leftrightarrow Y(z))
```
Further Formulas

\[ x < y \text{ ("x is lexicographically before y")}: \]
\[ \exists z (z_0 \preceq x \land z_1 \preceq y) \lor (x \preceq y \land x \neq y) \]

Finite(\(X\)):

“each subset \(Y\) of \(X\) has a minimal and a maximal element w.r.t. \(<\)”

\[ \forall Y \left( Y \subseteq X \land Y \neq \emptyset \right) \rightarrow \]
\[ \left( \exists y \text{ "y is } <\text{-minimal in } Y \" \land \exists y \text{ "y is } <\text{-maximal in } Y \" \right) \]
General Format of S2S-Formulas

A formula $\varphi(X_1, \ldots, X_n)$ defines a set of $\{0, 1\}^n$-labelled trees. Such a tree can be presented as a map $T : \{0, 1\}^* \rightarrow \{0, 1\}^n$.

Example:

$\varphi_0(X_1, X_2)$ might express

“$X_1$ is finite and each path sharing a $X_1$-element contains infinitely many $X_2$-elements”

All trees satisfying $\varphi_0$ form an S2S-definable tree language.

Rabin introduced tree automata equivalent to S2S.
Format of Tree Automata

\[ A = (Q, \Sigma, q_0, \Delta, \text{Acc}) \] where

\[ \Delta \subseteq Q \times \Sigma \times Q \times Q \]

A transition \((q, a, q_1, q_2)\) allows the automaton in state \(q\) at an \(a\)-labelled node \(u\) to proceed to states \(q_1, q_2\) at the two successor nodes \(u_0, u_1\)

A Büchi / Muller / parity tree automaton

\[ A = (Q, \Sigma, q_0, \Delta, F/F/c) \] accepts the tree \(t\) if there exists a run \(\varphi\) of \(A\) on \(t\) such that on each path of \(\varphi\) the acceptance condition is satisfied.
Example

\[ T_1 = \{ t \in T_{\{a,b\}}^\omega \mid \exists \text{path through } t \text{ with infinitely many } b \} \]

recognized by a Büchi tree automaton (and thus a parity tree automaton).

“Guess an appropriate path and on it check that infinitely often \( b \) occurs by visiting in the next step a final state.”

Use states \( q_a, q_b \) for the path to guessed, \( q_0 \) as initial state, and \( q \) for the other nodes.

\( q_0, q \) have color 0, \( q_a \) color 1, and \( q_b \) color 2.

Transitions: \((q_0, a/b, q_{a/b}, q), (q_0, a/b, q, q_{a/b})\),
the same with state \( q_a \) and state \( q_b \) in place of \( q_0 \),
\((q_a, a, q_{a/b}, q), (q_a, a, q, q_{a/b})\), similarly for \( q_b \),
finally \((q, a/b, q, q)\)
Example

A parity tree automaton recognizing

\[ T_2 = \{ t \in T_{\{a,b\}}^\omega \mid \text{each path through } t \text{ has only finitely many } b \} \]

Use \( q_a, q_b \) to signal input letters \( a, b \) respectively.

Define \( c(q_a) = 0, \quad c(q_b) = 1 \)

The maximal color occurring infinitely often on a path of the run is even (i.e., equal to 0) iff the letter \( b \) occurs only finitely often on the path.
Rabin’s Tree Theorem
Michael O. Rabin
(a) A tree language is definable in S2S iff it is recognizable by a parity tree automaton.

(b) The nonemptiness problem for parity tree automata “Given $\mathcal{A}$, does $\mathcal{A}$ accept some tree?” is decidable.

Consequence (from (b) for input-free tree automata):

Rabin’s Tree Theorem: $\text{MTh}(T_2)$ is decidable.

Everything works as before, but complementation and nonemptiness test are now more difficult.

We use positional determinacy of parity games.
Acceptance via Games

With any parity tree automaton $\mathcal{A} = (Q, \Sigma, q_0, \Delta, c)$ and any input tree $t$ associate a game $\Gamma_{\mathcal{A}, t}$ between two players “Automaton” and “Pathfinder” on the tree $t$
First Automaton picks a transition from $\Delta$ which can serve to start a run at the root of the input tree.

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 (compatible with the first transition and the input tree).

Then Pathfinder reacts again by branching left or right from the momentary node, etc.

Play gives a sequence of transitions (and hence a state sequence from $Q$), built up along a path chosen by Pathfinder.

Automaton wins the play iff the constructed state sequence satisfies the parity condition.
Game Positions

Positions of Automaton are the triples

$(\text{tree node } w, \text{tree label } t(w), \text{state } q \text{ at } w)$

By choice of a transition $\tau$ of the form $(q, t(w), q', q'')$, a position of Pathfinder is reached.

Positions of Pathfinder are the triples

$(\text{tree node } w, \text{tree label } t(w), \text{transition } \tau \text{ at } w)$

These positions with the moves define an infinite game graph.

Run Lemma: The tree automaton $\mathcal{A}$ accepts the input tree $t$ iff in the parity game $\Gamma_{\mathcal{A},t}$ there is a positional winning strategy for player Automaton from the initial position $(\varepsilon, t(\varepsilon), q_0)$
The Case of Input-Free Automata

We obtain a simpler game $\Gamma_\mathcal{A}$, ignoring where we are in the tree.

The game arena is then finite: It consists of

- the set $\Delta$ of transitions $(q, q', q'')$
- the states from $Q$

The winning condition is the parity condition.
Recall Results on Parity Games

- Parity games are positionally determined: From a given start position one of the two players has a winning strategy, which moreover is positional.

- For parity games over finite game graphs one can decide for any position who wins from this position.
Complementation of tree automata means to express the condition that a given automaton $\mathcal{A}$ does not accept $t$ by acceptance of another automaton.

Non-acceptance by $\mathcal{A}$ means non-existence of a winning strategy for Automaton in $\Gamma_{\mathcal{A},t}$.

Determinacy implies existence of a winning strategy for Pathfinder.

We convert this strategy into an automaton strategy in a different game $\Gamma_{\mathcal{B},t}$.

This gives the desired complement automaton $\mathcal{B}$.
Applying Determinacy (Step 1)

Proof: Let $\mathcal{A} = (Q, \Sigma, q_0, \Delta, c)$ be a parity tree automaton. We find a parity tree automaton $\mathcal{B}$ accepting precisely the trees $t \in T^\omega_\Sigma$ which are not accepted by $\mathcal{A}$

Start with the following equivalences: For any tree $t$,

$\mathcal{A}$ does not accept $t$

iff (by Run Lemma)

Automaton has no winning strategy from the initial position $(\varepsilon, t(\varepsilon), q_0)$ in the parity game $\Gamma_{\mathcal{A},t}$

iff (by Determinacy Theorem)

$(\ast)$ in $\Gamma_{\mathcal{A},t}$, Pathfinder has a positional winning strategy from $(\varepsilon, t(\varepsilon), q_0)$
Reformulate (*) in the form

“\( \mathcal{B} \) accepts \( t \)” for some tree automaton \( \mathcal{B} \)

Pathfinder’s strategy is a function \( f \) from the set \( \{0, 1\}^* \times \Sigma \times \Delta \) of his vertices into the set \( \{0, 1\} \) of directions.

Decompose this function into a family

\[
(f_w : \Sigma \times \Delta \rightarrow \{0, 1\})
\]

of “local instructions”, parameterised by \( w \in \{0, 1\}^* \)

The set \( I \) of possible local instructions \( i : \Sigma \times \Delta \rightarrow \{0, 1\} \) is finite,

Thus Pathfinder’s winning strategy can be coded by the \( I \)-labelled tree \( s \) with \( s(w) = f_w \)
Step 3

Let $s^\wedge t$ be the corresponding $(I \times \Sigma)$-labelled tree with

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

Now (*) is equivalent to the following:

There is an $I$-labelled tree $s$ such that for all sequences $\tau_0 \tau_1 \ldots$ of transitions chosen by Automaton and for all (in fact for the unique) $\pi \in \{0, 1\}^\omega$ determined by $\tau_0 \tau_1 \ldots$ via the strategy coded by $s$, the generated state sequence violates the parity condition.

This can be checked by a nondeterministic parity tree automaton $B$, the desired complement automaton for $A$. 

Wolfgang Thomas
The Input-free Case

An S2S-sentence without free variables will lead to an input-free tree automaton.

An input-free parity tree automaton $\mathcal{A} = (Q, q_0, \Delta, c)$ with $\Delta \subseteq Q \times Q \times Q$ defines the simpler game $\Gamma_\mathcal{A}$:

Automaton has positions in $Q$ and chooses transitions from $Q \times Q \times Q$

Pathfinder has positions in $\Delta$ and chooses directions.

Run Lemma (input-free case): $\mathcal{A}$ admits at least one successful run iff Automaton has a winning strategy in $\Gamma_\mathcal{A}$ from position $q_0$.

The first condition is checked effectively.

Wolfgang Thomas
\( \downarrow \) arrows define strategy of Automaton.
Rabin’s Tree Theorem

The theory $\text{MTh}(T_2)$ is decidable.

Proof

Consider an S2S-sentence $\varphi$

It can be transformed into an input-free parity tree automaton $A$ such that

the unlabelled infinite binary tree $T_2$ satisfies $\varphi$

iff $A$ has some successful run.

The second condition can be checked effectively.
Regular Trees
Rabin’s Basis Theorem

Recall: A nonempty regular $\omega$-language contains an ultimately periodic $\omega$-word.

A corresponding result holds for nonempty tree languages which are recognized by parity tree automata.

Rabin’s Basis Theorem: A nonempty tree language recognized by a parity tree automaton contains a regular tree.

A tree $t \in T^\omega_\Sigma$ is called regular if it is “finitely generated” in the following sense:

There is a deterministic finite automaton equipped with output which tells for any given input $w \in \{0, 1\}^*$ which label is at node $w$ (i.e. the value $t(w)$).
Examples

Wolfgang Thomas
Rabin’s Basis Theorem: Proof

Assume \( A = (Q, \Sigma, q_0, \Delta, c) \) is a parity tree automaton.

Proceed to an “input-guessing” (and input-free) tree automaton \( A' \) with states in \( Q \times \Sigma \):

\( A' \) guesses an input tree and works on it as \( A \) does.

\( A' \) may have several initial states.

Then:

The input-free automaton \( A' \) admits a successful run iff \( T(A) \neq \emptyset \), and a tree in \( T(A) \) is extracted from the second components of the run.

Thus a regular tree is generated.

via winning strategy of Automaton

Wolfgang Thomas
Büchi automata, Muller automata, and parity tree automata provide different versions of quantifier elimination:

to $\Sigma_1^1$, to $\text{Bool}(\Pi_2^0)$.

Tree automata provide a less radical way of quantifier elimination than Büchi automata:

An S2S-formula $\varphi(X_1, \ldots, X_n)$ can be transformed into a formula with two second-order quantifiers:

“There is a run on the tree given by $X_1, \ldots, X_n$ such that on each path the acceptance condition is satisfied.”

In logical terminology this is a $\Sigma_2^1$-condition.