Automata on Infinite Trees

Wolfgang Thomas

RWTH AACHEN

Edinburgh, October 2014

MSO-Logic over the Infinite Binary Tree

The Model T_2

The structure of the infinite binary tree is

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

where S_i is the i-th successor function:

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

The monadic second-order theory (short: the monadic theory) of the infinite binary tree, denoted by $\mathrm{MTh}_2(\underline{T_2})$

is the set of MSO-sentences true in T_2.

A Σ-labelled binary tree is presented as $T : \{0,1\}^* \to \Sigma$.

Example Formulas

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

$\varphi_s^*(x,y)$ **with** $\varphi_s(z,z') := z0 = z' \vee z1 = z'$

$\forall X\big((X(y) \wedge \forall z(X(z0) \to X(z)) \wedge \forall z(X(z1) \to X(z))) \to X(x)\big)$

$\mathrm{Chain}(X)$ **("X is linearly ordered by $\preceq$"):**

$\forall x \forall y((X(x) \wedge X(y)) \to (x \preceq y \vee y \preceq x))$

$\mathrm{Path}(X)$ **("X is a path, i.e. a maximal chain"):**

$\mathrm{Chain}(X) \wedge \neg\exists Y(X \subseteq Y \wedge X \neq Y \wedge \mathrm{Chain}(Y))$

$X \subseteq Y$: $\forall z(X(z) \to Y(z))$

$X = Y$: $\forall z(X(z) \leftrightarrow Y(z))$

Further Formulas

$x < y$ **("x is lexicographically before y"):**

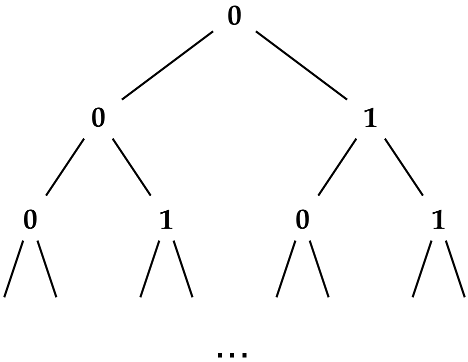
$$\exists z(z0 \preceq x \wedge z1 \preceq y) \vee (x \preceq y \wedge x \neq y)$$

$\mathrm{Finite}(X)$**:**

"each subset Y of X has a minimal and a maximal element w.r.t. $<$"

$$\forall Y(\ (Y \subseteq X \wedge Y \neq \emptyset) \rightarrow$$
$$(\exists y \text{ "}y\textbf{ is }<\textbf{-minimal in }Y\text{"} \wedge \exists y \text{ "}y\textbf{ is }<\textbf{-maximal in }Y\text{"}\ ))$$

A Labelled Tree



MSO-Definable Tree Languages

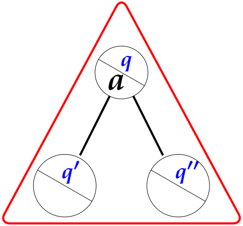
The set of $\{0,1\}$-labelled trees having a path with infinitely many occurrences of label 1.

The set of $\{0,1\}$-labelled trees with only finitely many labels 1.

The set of trees with label 1 on precisely three infinite paths

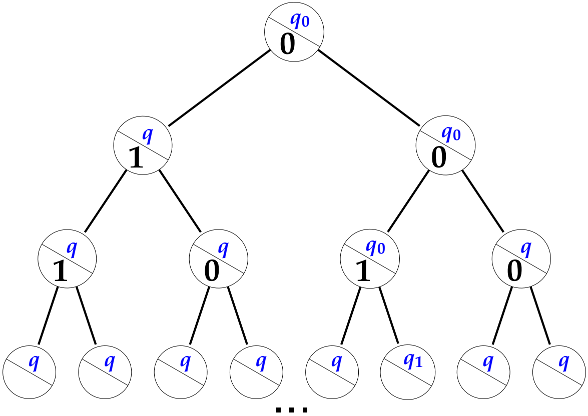
Tree Automata

Transition of a Tree Automaton



$$(q, a, q', q'')$$

A Run



Format of Tree Automata

$\mathcal{A} = (Q, \Sigma, q_0, \Delta, \mathrm{Acc})$ 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 $u0, u1$

A Büchi / Muller / parity tree automaton $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F/\mathcal{F}/c)$ accepts the tree t

if there exists a run ϱ of $\mathcal{A}$ on t such that on each path of ϱ the acceptance condition is satisfied.

On Quantifier Elimination

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

from Σ_n^1 to Σ_1^1, and then to $\mathrm{Bool}(\Pi_2^0)$.

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

An MSO-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 Y the acceptance condition is satisfied."

In logical terminology this is a Σ_2^1-condition.

Example

$T_1 = \{t \in T^{\omega}_{\{0,1\}} \mid$
exists path through t with infinitely many $1\}$

recognized by a Büchi tree automaton:

Guess an appropriate path and on it check that infinitely often 1 occurs by visiting in the next step a final state.

Example

A parity tree automaton recognising

$T_2 = \{t \in T^{\omega}_{\{0,1\}} \mid$
each path through t has only finitely many $1\}$

Use q_0, q_1 to signal input letters $0, 1$ respectively.

Define $c(q_0) = 0,\ c(q_1) = 1$

The maximal color occurring infinitely often on a path of the run is even (i.e., equal to 0) iff the letter 1 occurs only finitely often on the path.

This tree language is NOT recognizable by a Büchi tree automaton!

Rabin’s Tree Theorem

Rabin's Tree Theorem

A tree language over an alphabet $\{0,1\}^n$ is MSO-definable iff it is recognizable by a parity tree automaton.

The MSO-theory of T_2 is decidable.

Everything works as before for Büchi automata, but complementation and emptiness test are now more difficult.

Complementation via Games

- **Characterize "$\mathcal{A}$ accepts t":**
  In a two-player game $\Gamma(\mathcal{A}, t)$ between "Automaton" and "Pathfinder", "Automaton" has a winning strategy.

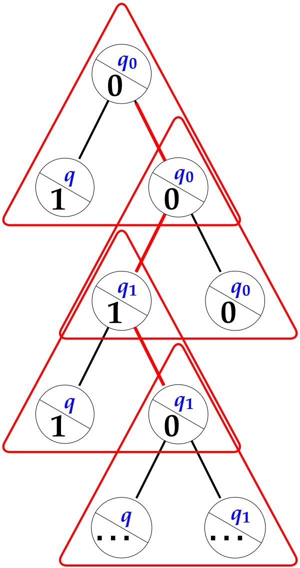
- **Non-acceptance of t by $\mathcal{A}$ means non-existence of a winning strategy for Automaton in the game $\Gamma(\mathcal{A}, t)$.**

- **Apply determinacy : There is a winning strategy for Pathfinder.**

- **Convert this strategy into an automaton strategy in a different game $\Gamma(\mathcal{B}, t)$.**

- **This gives the desired complement automaton $\mathcal{B}$.**

A play of the game $\Gamma(\mathcal{A}, t)$



$\Gamma(\mathcal{A}, t)$

First Automaton picks a transition from Δ 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

(tree node w, tree label $t(w)$, state q at w)

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

Positions of Pathfinder are the triples

(tree node w, tree label $t(w)$, transition τ at w)

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 $(\varepsilon, t(\varepsilon), q_0)$

Recall Results on Parity Games

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

2. The set of positions of a parity game graph from which a given player wins is MSO-definable (in the MSO-language for game graphs).

3. For parity games over finite game graphs one can decide for any position who wins from this position.

Complementation (Step 1)

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 $t \in T_\Sigma^\omega$ 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)

(*) in $\Gamma(\mathcal{A}, t)$, Pathfinder has a positional winning strategy from $(\varepsilon, t(\varepsilon), q_0)$

Step 2

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

$$(f_w : \Sigma \times \Delta \to \{0,1\})$$

of "local instructions", parameterised by $w \in \{0,1\}^*$

The set I of possible local instructions $i : \Sigma \times \Delta \to \{0,1\}$ 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

$s^\wedge t(w) = (s(w), t(w))$ 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.

More Detail

A reformulation of this yields:

(1) There is an I-labelled tree s such that $s^\wedge t$ satisfies:

(2) for all $\pi \in \{0,1\}^\omega$

(3) for all $\tau_0\tau_1 \ldots \in \Delta^\omega$

(4) if the sequence $s|\pi$ of local instructions applied to the sequence of tree labels $t|\pi$ and to the transition sequence $\tau_0\tau_1 \ldots$ indeed produces the path π, then the state sequence determined by $\tau_0\tau_1 \ldots$ violates the parity condition.

Condition (4) describes a property of ω-words over $I \times \Sigma \times \Delta \times \{0,1\}$

which obviously can be checked by a sequential parity automaton $\mathcal{M}_4$, independently of t.

(1) There is an I-labelled tree s such that $s^\wedge t$ satisfies:

(2) for all $\pi \in \{0,1\}^\omega$

(3) for all $\tau_0\tau_1 \ldots \in \Delta^\omega$

(4) if the sequence $s|\pi$ of local instructions applied to the sequence of tree labels $t|\pi$ and to the transition sequence $\tau_0\tau_1 \ldots$ indeed produces the path π, then the state sequence determined by $\tau_0\tau_1 \ldots$ violates the parity condition.

Condition (3) describes a property of ω-words over $I \times \Sigma \times \{0,1\}$, which results from (4) by a universal quantification (equivalently, by a negation, a projection, and another negation).

(3) is checked by a sequential and deterministic Muller automaton $\mathcal{M}_3$

(1) There is an I-labelled tree s such that $s^\wedge t$ satisfies:

(2) for all $\pi \in \{0,1\}^\omega$

(3) for all $\tau_0\tau_1 \ldots \in \Delta^\omega$

(4) if the sequence $s|\pi$ of local instructions applied to the sequence of tree labels $t|\pi$ and to the transition sequence $\tau_0\tau_1 \ldots$ indeed produces the path π, then the state sequence determined by $\tau_0\tau_1 \ldots$ violates the parity condition.

Condition (2) defines a property of $(I \times \Sigma)$-labelled trees, which can be checked by a deterministic Muller *tree* automaton $\mathcal{M}_2$, simulating $\mathcal{M}_3$ along each path.

(Note that, by determinism of $\mathcal{M}_3$, the $\mathcal{M}_3$-runs on different paths of an $(I \times \Sigma)$-labelled tree agree on the respective common prefix and hence can be merged into one run of $\mathcal{M}_2$)

(1) There is an I-labelled tree s such that $s^\wedge t$ satisfies:

(2) for all $\pi \in \{0,1\}^\omega$

(3) for all $\tau_0\tau_1\ldots \in \Delta^\omega$

(4) if the sequence $s|\pi$ of local instructions applied to the sequence of tree labels $t|\pi$ and to the transition sequence $\tau_0\tau_1\ldots$ indeed produces the path π, then the state sequence determined by $\tau_0\tau_1\ldots$ violates the parity condition.

Applying nondeterminism, a Muller tree automaton $\mathcal{B}$ can be built which checks Condition (1), by guessing a tree s on the input tree t and working on $s^\wedge t$ like $\mathcal{M}_2$

$\mathcal{B}$ does not depend on the tree t under consideration.

Thus $\mathcal{B}$ accepts precisely those trees which $\mathcal{A}$ does not accept, as was to be shown.

Intermediate Summary

We have completed the translation from MSO-logic to tree automata:

An MSO-definable tree language over an alphabet $\{0,1\}^n$ is recognizable by a parity tree automaton.

We now show

The MSO-theory of T_2 is decidable.

Let φ be an MSO-sentence.

Use the above result for $n = 0$ and obtain a parity tree automaton $\mathcal{A}_\varphi$ with unlabelled transitions (q, q', q'').

We have to check whether $\mathcal{A}_\varphi$ has a successful run.

The Input-free Case

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

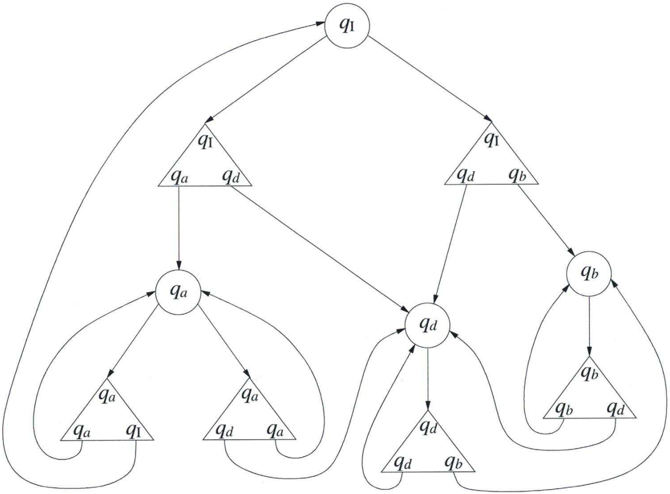
Automaton has positions in in Q and chooses transitions from $Q \times Q \times Q$

Pathfinder has positions in Δ and chooses directions.

The corresponding game graph is finite!

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 by the 3rd result on parity games.


q_I
q_I
q_a
q_d
q_I
q_d
q_b
q_b
q_a
q_d
q_b
q_a
q_a
q_I
q_a
q_d
q_a
q_d
q_d
q_b
q_b
q_b
q_d

Regular Trees

Definition

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 $t(w)$).

Rabin's Basis Theorem

Recall:

A nonempty regular ω-language contains an ultimately periodic ω-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.

Rabin's Basis Theorem: Proof

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

Proceed to an "input-guessing" (and input-free) tree automaton $\mathcal{A}'$ with states in $Q \times \Sigma$:

$\mathcal{A}'$ guesses an input tree and works on it as $\mathcal{A}$ does.

$\mathcal{A}'$ may have several initial states.

Then:

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

Thus a regular tree is generated.

Rabin's Solution of Church's Problem

The Tree Setting

Consider moves of Players 1 and 2 in $\{0,1\}$.

The players construct a labelled path in T_2.

A dummy value is associated to the root (say 0).

Player 1 chooses directions.

Player 2 chooses a label at a node reached by Player 1.

When Player 1 has chosen the bit sequence w, Player 2 puts his choice as label at position w.

So a strategy of Player 2 is a $\{0,1\}$-labelling of T_2.

Defining Winning Strategies in MSO

A play is a sequence "direction-label-direction-label etc."

The winning condition is a condition on labelled paths through T_2.

If the winning condition definable by an MSO-formula $\varphi(X, Y)$, one can reformulate it as an MSO condition on paths through the tree.

Consequence: There is an MSO-formula $\psi(Z)$ expressing:

The tree labelling given by Z defines a winning strategy for Player 2.

("For all paths, the sequence Y of directions and the sequence X of Z-labels along Y satisfies $\varphi(X, Y)$")

Application of Tree Automata

- By **Rabin's Tree Theorem**,
  one can decide truth of $\exists Z \psi(Z)$,
  so can decides whether Player 2 wins the game
  defined by φ.

- In this case, by **Rabin's Basis Theorem**
  a regular tree exists satisfying $\psi(Z)$, which gives a
  finite-state strategy for Player 2.

The problem of solving sequential games is a satisfiability problem over trees.