Tree Automata with Tests and Predicate Logic

Ralf Treinen

LRI

Université Paris-Sud

Orsay, France

Plan of the talk

Preliminaries

- Tree automata (*TA*)
- The logic *WS2S*
- Equivalence of *TA* and *WS2S*
- Tree automata with tests (*TAT*)

Contribution of this talk

- Tree automata with component-wise tests (*TACT*)
- Emptiness of *TACT* is undecidable
- Logical characterisation of *TAT* and *TACT*
- Open problems

1 Tree Automata

[Doner, Thatcher, Wright: 1965-1970]

A tree automaton (TA) is given by

- a tree signature Σ
- a finite set Q of states
- a set $F \subseteq Q$ of final states
- a set Δ of transition rules of the form

$$f(q_1, q_2) \to q$$

Example:

- $\Sigma = \{0/0, s/1, nil/0, cons/2\}$
- $Q = \{N, L\}$
- $F = \{L\}$
- $\Delta = \left\{ \begin{array}{lllll} 0 & \to N & \quad s(N) & \to N \\ nil & \to L & \quad cons(L, N) & \to L \end{array} \right\}$

The move relation $\rightarrow_{\mathcal{A}}$ of an automaton $\mathcal{A}$ is the smallest relation with

- $a \rightarrow_{\mathcal{A}} q(a)$ if $q \rightarrow a \in \Delta$
- if $t_1 \rightarrow_{\mathcal{A}} q_1(t_1)$

  $t_2 \rightarrow_{\mathcal{A}} q_2(t_2)$

  $f(q_1, q_2) \rightarrow q \in \Delta$

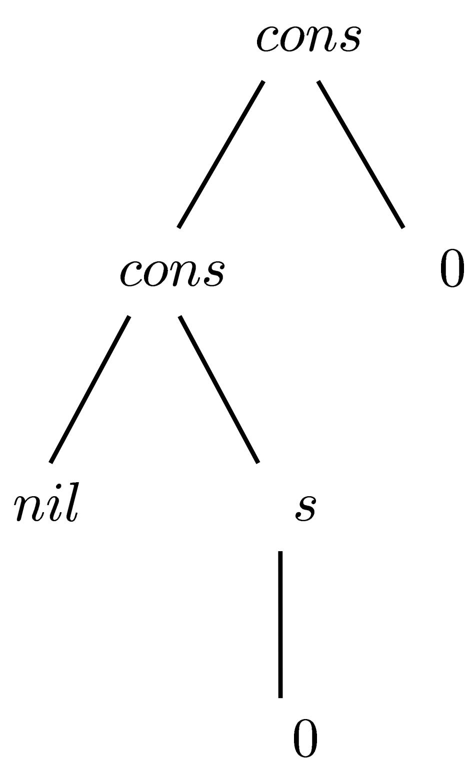
  then $f(t_1, t_2) \rightarrow_{\mathcal{A}} q(f(t_1, t_2))$

The automaton $\mathcal{A}$ accepts the tree t if $t \rightarrow_{\mathcal{A}} q(t)$ for some $q \in F$.

The language recognised by $\mathcal{A}$ is

$$L(\mathcal{A}) = \{t \mid \mathcal{A} \text{ accepts } t\}$$

1.1 Example



Transition rules:

$$\begin{array}{lll} 0 \to N & \quad s(N) & \to N \\ nil \to L & \quad cons(L,N) & \to L \end{array}$$

$$N(\quad)$$

$L(\quad) \qquad N(\ )$

$$L(\qquad) \qquad N(\;)$$

$$L(\qquad)$$

1.2 Properties of Tree Automata

Tree automata enjoy all nice properties of word automata:

- closure under $\cup$, $\cap$, complement, linear homomorphisms and inverse homomorphisms
- decidability of emptiness
- determinisation
- minimisation

2 The logic *WS2S*

2.1 Definition

Predicate logic language and its standard model:

	Syntax	*Semantics*
Sorts	$\mathtt{w}$	$\{1,2\}^*$
	$\mathtt{s}$	the finite subsets of $\{1,2\}^*$
Functions	$\epsilon\colon \mathtt{w}$	ϵ
	$1\colon \mathtt{w} \to \mathtt{w}$	$x \mapsto x1$
	$2\colon \mathtt{w} \to \mathtt{w}$	$x \mapsto x2$
Predicates	$\in\colon \mathtt{w} \times \mathtt{s}$	$\in$

2.2 Examples

Some relations definable in *WS2S*:

$$
\begin{aligned}
X \subseteq Y \quad &:\Leftrightarrow \quad \forall z(z \in X \rightarrow z \in Y) \\
X = Y \cup Z \quad &:\Leftrightarrow \quad \forall v(v \in X \leftrightarrow (v \in Y \vee v \in Z)) \\
x \ \textit{prefix-of}\ y \quad &:\Leftrightarrow \quad \forall Z(y \in Z \\
&\qquad\quad \wedge \forall v(v1 \in Z \vee v2 \in Z \rightarrow v \in Z) \\
&\qquad\quad \rightarrow x \in Z) \\
\textit{even(x)} \quad &:\Leftrightarrow \quad \exists E, O\Big(\epsilon \in E \wedge x \in E \\
&\qquad\quad \wedge E \cap O = \emptyset \\
&\qquad\quad \wedge \forall z(z1 \in E \vee z2 \in E \rightarrow z \in O) \\
&\qquad\quad \wedge \forall z(z1 \in O \vee z2 \in O \rightarrow z \in E)\Big)
\end{aligned}
$$

3 Equivalence of TA and *WS2S*

Theorem [Thatcher & Wright 68]:

- Tree Automata and *WS2S* have the same expressive power.
- Validity on *WS2S* is decidable.

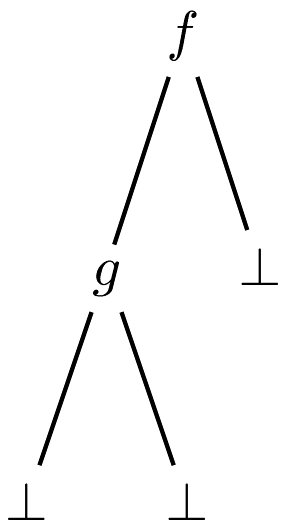
A Tree Automaton describes a set of trees. A formula $\Phi(X_1, \ldots, X_n)$ describes a set of n-tuples of sets of words.

What is the exact correspondence between these two categories?

3.1 Translating TA into *WS2S*

Let t be a tree over the signature $\{f_1, \ldots, f_n\}$. Its tuple representation $\bar{t}$ is $\{M_1 \ldots, M_n\}$, $M_i \subseteq \{1,2\}^*$, where M_i is the set of positions of t that are marked with f_i.

Example: Signature $= \{f, g, \perp\}$



$$(\,\{\epsilon\}\,,\,\{1\}\,,\,\{11,12,2\}\,)$$

Lemma: For every Tree Automaton $\mathcal{A}$ over $\{f_1, \ldots, f_n\}$ there is a formula $\Phi_{\mathcal{A}}(X_1, \ldots, X_n)$ such that

$$\mathcal{A} \text{ accepts } t \qquad \Leftrightarrow \qquad \bar{t} \models \Phi_{\mathcal{A}}(X_1, \ldots, X_n)$$

$$
\begin{aligned}
partition(X, X_1, \ldots, X_n) &\Leftrightarrow X = X_1 \cup \ldots \cup X_n \wedge \bigwedge_{i \neq j} X_i \cap X_j = \emptyset \\
term(X, X_1, \ldots, X_n) &\Leftrightarrow partition(X, X_1, \ldots, X_n) \\
&\quad \forall x, y (x \leq y \wedge y \in X \rightarrow x \in X) \\
&\quad \forall x (\bigwedge_{f_i \text{ binary}} x \in X_i \rightarrow x1 \in X \wedge x2 \in X) \\
&\quad \forall x (\bigwedge_{f_i \text{ constant}} x \in X_i \rightarrow x1 \notin X \wedge x2 \notin X)
\end{aligned}
$$

(Here: Signature contains only binary and constant symbols).

States: $\{q_1, \ldots, q_k\}$

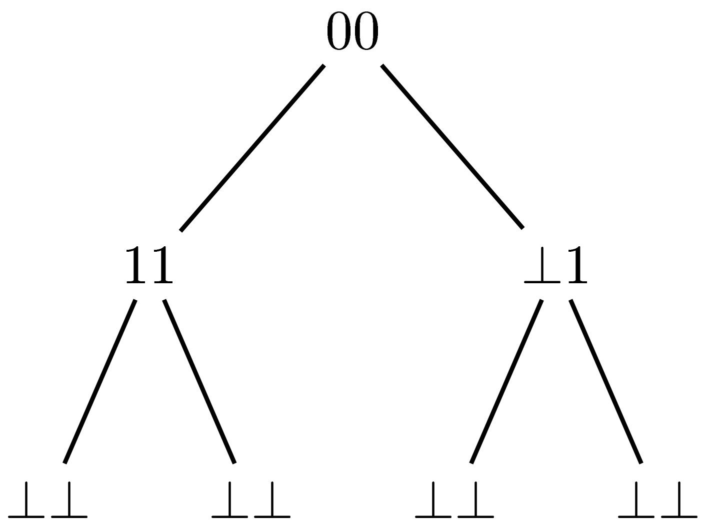
$$
\begin{aligned}
\phi_A(X_1, \ldots, X_n) \quad &= \exists X, Q_1, \ldots, Q_k \Big(\\
&\qquad term(X, X_1, \ldots, X_n) \\
&\wedge \quad partition(X, Q_1, \ldots, Q_k) \\
&\wedge \quad \bigvee_{q_i \in F} \epsilon \in Q_i \\
&\wedge \quad \forall x(\bigwedge_{f_i \text{ constant}} x \in X_i \rightarrow \bigvee_{f_i \rightarrow q_j \in \Delta} x \in Q_j) \\
&\wedge \quad \forall x(\bigwedge_{f_i \text{ binary}} x \in X_i \rightarrow \bigvee_{f_i(q_1, q_2) \rightarrow q_j \in \Delta} x \in Q_j \wedge x_1 \in Q_1 \wedge x_2 \in Q_2) \Big)
\end{aligned}
$$

3.2 Translating *WS2S* into TA

Let $p = (M_1, \ldots, M_n) \in \{1,2\}^* \times \ldots \times \{1,2\}^*$. Its tree representation is $\bar{p}$:

- Signature: $\perp^n$ (constant), $\{\perp, 0, 1\}^n$ (all binary)
- Set of positions: prefix closure of $(M_1 \cup \ldots \cup M_n)\{1,2\}$
- Component i marked 1 iff $\pi \in M_i$, marked 0 iff $\pi \notin M_i$ and some extension of π is in M_i, $\perp$ otherwise.

$(\{1\}, \{1,2\})$



Lemma: For every formula $\Phi(X_1, \ldots, X_n)$ there is an automaton $\mathcal{A}_\Phi$ such that

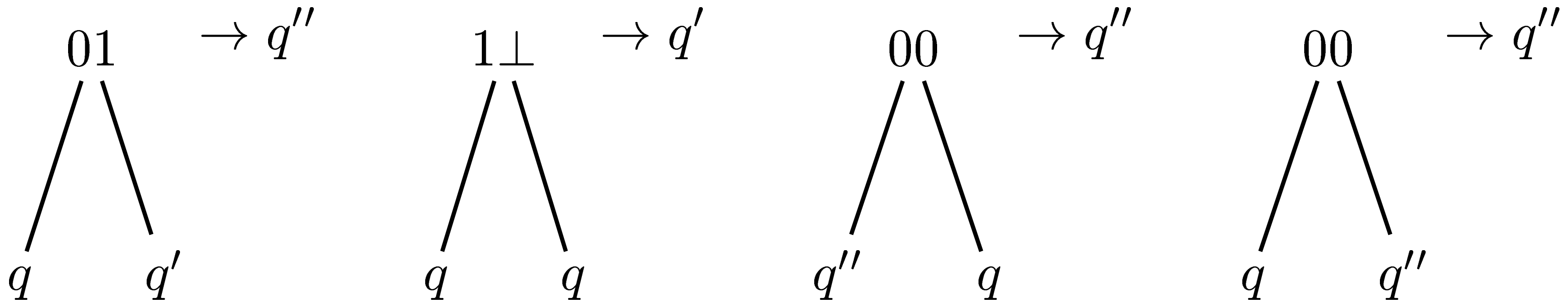
$$p \models \Phi(X_1, \ldots, X_n) \qquad \Leftrightarrow \qquad \mathcal{A}_\Phi \text{ accepts } \bar{p}$$

Simplification:

- only set variables
- atomic formulae: $X \subseteq Y$, $X = \epsilon$, $X = Y.1$, $X = Y.2$.

first step: automata for the atomic formulae. Example: $X = Y.2$:

$$\perp\perp \to q$$



Inductive construction using the closure properties of tree automata:

- conjunction $\Rightarrow$ intersection
- negation $\Rightarrow$ complement (using determinisation)
- existential quantification $\Rightarrow$ projection

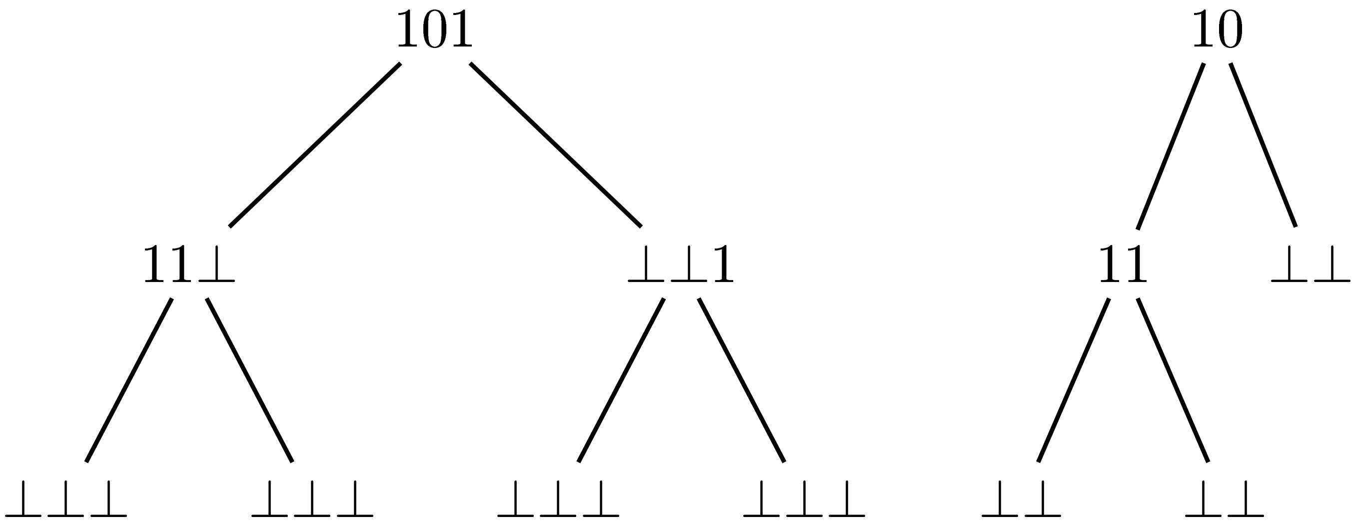
Plus cylindrification:

To build the conjunction of $\Phi(X_1, X_2)$ and $\Psi(X_2, X_3)$

- Build $\mathcal{A}_\Phi$ and $\mathcal{A}_\Psi$
- Cylindrify to the variables (X_1, X_2, X_3)
- Intersection automaton

Projection of a Tree

Example: 3rd component:



4 Tree Automata with Tests (TAT)

[Mongy and Dauchet, early 80's]: Have transitions with tests.

Example:

$$a \to q \qquad f(q,q) \to^{12=12} q$$

Emptiness undecidable even when only comparisons between cousin positions [Seynhaeve et. al. 97]

Restriction to tree automata with tests between brothers [Bogaert & Tison 92]:

- Closure under Boolean operations
- Emptiness decidable.

Example:

$$a \to q \quad f(q,q) \to^{1=2} q$$

defines the complete binary trees.

What is a corresponding logic ?

5 Tree Automata with Component-wise Tests (TACT)

If we want to translate any logical formula of a logic (some extension of *WS2S*) into automata then we need closure under cylindrification.

TACT: Tests on Components:

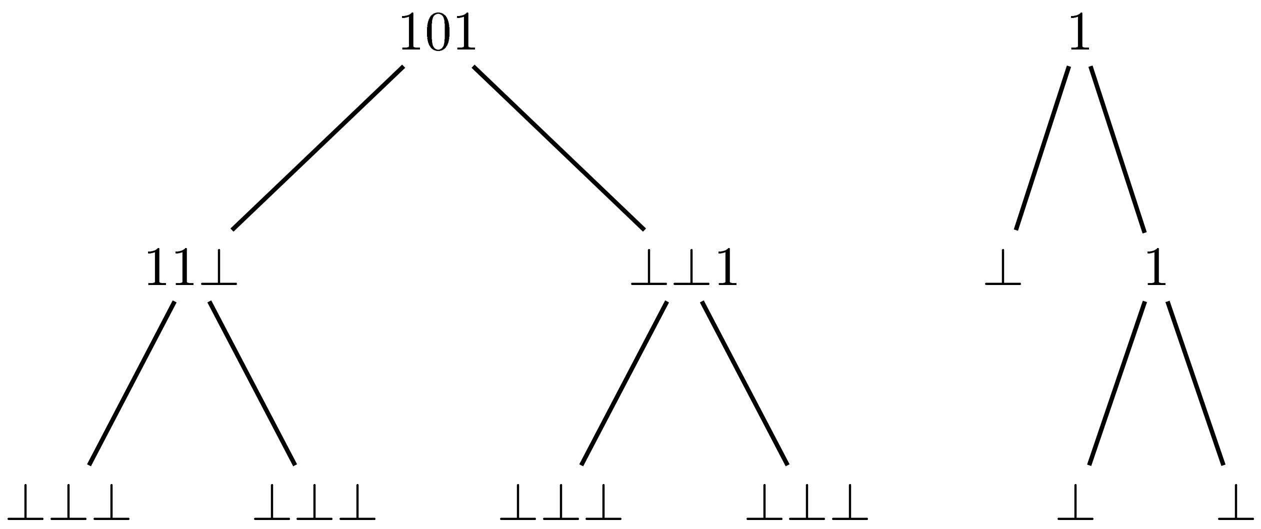
$$f(q_1, q_2) \rightarrow^{1.i=2.i} q$$

In general: any boolean combination of tests:

$$f(q_1, q_2) \rightarrow^{1.i=2.i \vee (1.j \neq 2.j \wedge 1.k=2.k)} q$$

Component of a Tree

Example 3rd component:



6 Emptiness of TACT is Undecidable

Reduction of PCP:

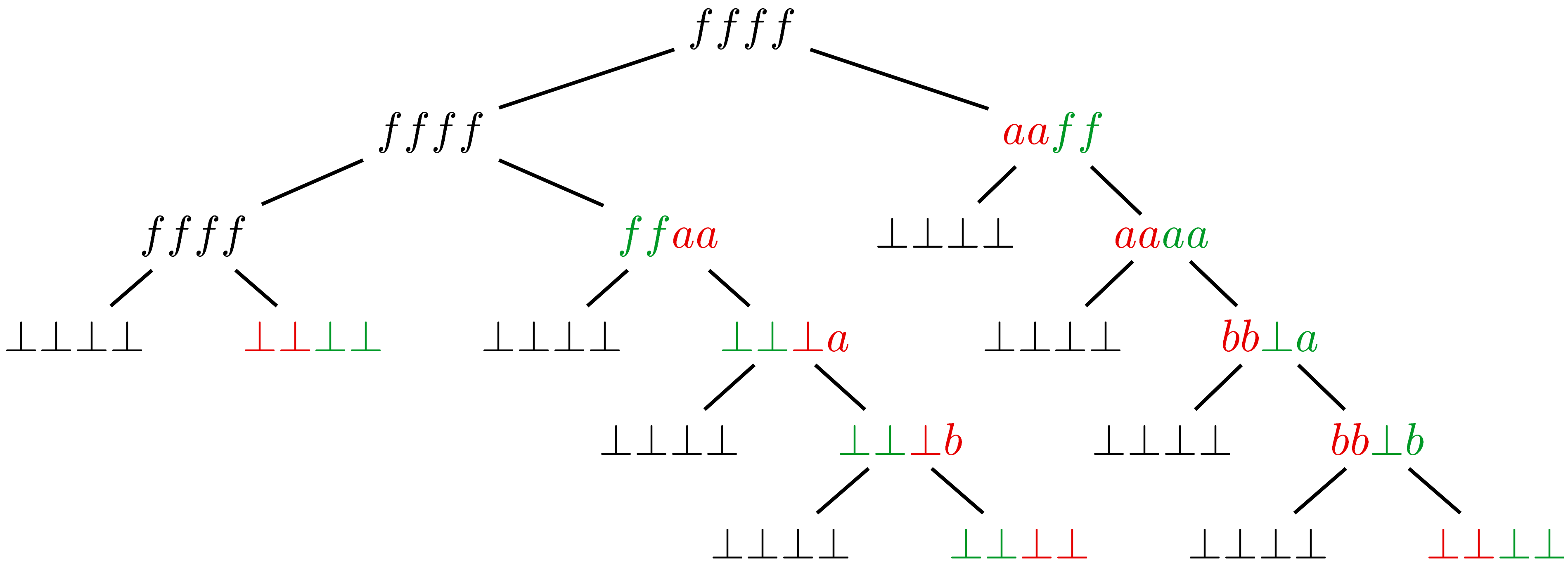
An instance of the Post Correspondence Problem (short: PCP) is a finite sequence $P = ((p_i, q_i))_{i=1,\dots,m}$ of pairs of words from $\{a, b\}^*$.

A solution of such an instance of PCP is a nonempty sequence $(i_1, \dots, i_n)$, $1 \leq i_j \leq m$, such that $p_{i_1} \cdots p_{i_n} = q_{i_1} \cdots q_{i_n}$.

It is undecidable whether an instance of the PCP has a solution or not [Post 46].

Instance: $((a, aab), (abb, b))$.

Solution: $((\epsilon, \epsilon), (a, aab), (aabb, aabb))$



For any word $v \in \Gamma^*$, let $A_v^{i,j}$ for $i \neq j$ be the automaton that accepts t iff t is a comb, $t\downarrow_i = fw$ and $t\downarrow_j = wv$ for some $w \in \Gamma^*$.

Let $v = v_1 \cdots v_n$:

$$\begin{array}{rcll}
(\perp\perp\perp\perp) & \to & q_l & \\
(\perp\perp\text{__})(q_l, q_l) & \to & q_n & \\
(\perp\perp\text{__})(q_l, q_n) & \to & q_n & \\
(\perp v_i\text{__})(q_l, q_i) & \to & q_{i-1} & i = 2, \ldots, n \\
(f v_1\text{__})(q_l, q_1) & \to & q_0 & \\
(\alpha v_1\text{__})(q_l, q_1) & \to & q_\alpha & \\
(\alpha\beta\text{__})(q_l, q_\beta) & \to & q_\alpha & \\
(f\beta\text{__})(q_l, q_\beta) & \to & q_0 &
\end{array}$$

The only accepting state is q_β.

Define A_P^+ that accepts t iff t is a comb and for some i we have that $t\downarrow_1 = fw_1$, $t\downarrow_2 = fw_2$, $t\downarrow_3 = w_1 p_i$ and $t\downarrow_4 = w_2 q_i$.

Define A_P^- analogously.

Define AE_P^+ and AE_P^- that test in addition if the newly constructed pair is of the form (v, v).

Finally:

$$
\begin{array}{rlll}
(\bot\bot\bot\bot) & \to & p_\bot & \\
(ffff)(p_\bot, p_\bot) & \to & p^+ & \\
(ffff)(p^+, q^+) & \to^{1.1=2.1 \wedge 1.2=2.2} & p^- & (q^+ \in Q^+) \\
(ffff)(p^-, q^-) & \to^{1.3=2.3 \wedge 1.4=2.4} & p^+ & (q^- \in Q^-) \\
(ffff)(p^+, q^+) & \to^{1.1=2.1 \wedge 1.2=2.2} & p_f & (q^+ \in Q^+_=) \\
(ffff)(p^-, q^-) & \to^{1.3=2.3 \wedge 1.4=2.4} & p_f & (q^- \in Q^-_=)
\end{array}
$$

where p_f is the only accepting state.

7 The Logic *WS2Sy*

WS2S plus synchronisation predicate: $Sy(x, X)$.

$$w, M \models Sy(x, X) \quad \Leftrightarrow \quad \forall v (w1v \in M \leftrightarrow w2v \in M)$$

$Sy(x, X_1, \ldots, X_n)$ for $Sy(x, X_1) \wedge \ldots \wedge Sy(x, X_n)$.

1. restricted *WS2Sy*: if Φ contains $Sy(x, X_1, \ldots, X_n)$ then X_i is a free variable.
2. uniform *WS2Sy*: if Φ contains $Sy(x, X_1, \ldots, X_n)$ then $\{X_1, \ldots, X_n\}$ is the set free variables.

Theorem:

- TAT have the same expressive power than uniform *WS2Sy*
- TACT have the same expressive power than restricted *WS2Sy*

7.1 Translating TACT into restricted *WS2Sy*

OBdA: all tests are conjunctions of positive and negative equations.

$$
\begin{aligned}
&\dots\\
\wedge \quad & \bigwedge_{(i_1\dots i_n)\neq(0\dots0)} \forall x\big(x \in X_{i_1} \cap \dots \cap X_{i_n} \rightarrow \bigvee_{(i_1\dots i_n)(q_j,q_k)\rightarrow^c q_l\in\Delta} (\\
& \quad x \in Q_l \wedge x1 \in Q_j \wedge x2 \in Y_k\\
& \quad \wedge \bigwedge_{1.i=2.i\in c} Sy(x, X_0^i, \dots, X_m^i) \wedge \bigwedge_{1.i\neq2.i\in c} \neg Sy(x, X_0^i, \dots, X_m^i)))\big)
\end{aligned}
$$

Special case TAT: uniform *WS2Sy*

7.2 Translating uniform *WS2Sy* to TACT

Automaton for $Sy(X_1, X_2, \ldots, X_n)$:

$$
\begin{array}{rcl}
(\perp \ldots \perp) & \rightarrow & q_0 \\
(\perp \_ \ldots \_)(q_0, q_0) & \rightarrow & q_0 \\
(0 \_ \ldots \_)(q_0, q_0) & \rightarrow & q_0 \\
(1 \_ \ldots \_)(q_0, q_0) & \rightarrow^{1.2=2.2 \wedge \ldots \wedge 1.n=2.n} & q_1 \\
(0 \_ \ldots \_)(q_0, q_1) & \rightarrow & q_1 \\
(0 \_ \ldots \_)(q_1, q_0) & \rightarrow & q_1
\end{array}
$$

Closure by intersection since the set of tests is closed under conjunction.

Closure by negation

Needs determinisation.

$$\begin{array}{lll} f(q_1, q_2) & \rightarrow^{c} & q \\ f(q_1, q_2) & \rightarrow^{d} & p \end{array}$$

dertminises to

$$\begin{array}{lll} f(q_1, q_2) & \rightarrow^{c \wedge d} & \{p, q\} \\ f(q_1, q_2) & \rightarrow^{c \wedge \neg d} & \{q\} \\ f(q_1, q_2) & \rightarrow^{\neg c \wedge d} & \{p\} \end{array}$$

Closure by existential quantification

Given: automaton with transitions Δ.

Existential quantification of the last component yields the set of transitions

$$\begin{array}{rcl}\Delta' & = & \{(\bot\cdots\bot) \to q \mid (\bot\cdots\bot\bot) \to q \in \Delta\} \\ & \cup & \{(\bot\cdots\bot) \to q \mid (\bot\cdots\bot f)(q_1, q_2) \to q \in \Delta, f \neq \bot\} \\ & \cup & \{\bar{f}(q_1, q_2) \to^c q \mid \bar{f} f(q_1, q_2) \to^c q, \bar{f} \neq \bar{\bot}\}\end{array}$$

Closure by cylindrification

Given: automaton over n components with transitions Δ.

Cylindrification by the $n+1$-th component yields the set of transitions

$$\begin{aligned}
\Delta' &= \{(\perp\cdots\perp\perp)\to q \mid (\perp\cdots\perp)\to q\in\Delta\}\\
&\cup \{(\bar{f}f)(q_1,q_2)\to^c q \mid \bar{f}(q_1,q_2)\to^c q\in\Delta, f\neq\perp,\ \bar{f}\neq\bar{\perp}\}\\
&\cup \{(\bar{\perp}f)(q_1,q_2)\to q \mid (\perp\cdots\perp)\to q\in\Delta, f\neq\perp\}
\end{aligned}$$

The case of uniform formulae

For all operations on TACT: one obtains new tests only by Boolean combinations.

Hence, the automaton finally obtained has only the two tests

- $1.1 = 2.1 \wedge \cdots \wedge 1.n = 2.n$ (equivalent to $1 = 2$)
- $1.1 \neq 2.1 \vee \cdots \vee 1.n \neq 2.n$ (equivalent to $1 \neq 2$)

hence it is a TAT.

8 Summary

- TA = $WS2S$ [Thatcher & Wright]
- TAT = uniform $WS2Sy$
- TACT = restricted $WS2Sy$

9 Open Problems

- TACT closed under arbitrary projections?
- What's about (generalised) reduction automata [Caron et. al. 93, 94]