

In order to define A_M for positive example $M \in P$, we will use a more succinct automaton which can be compiled to a (bottom-up) tree automaton at the cost of an exponential increase in size, analogous to the subset construction for determinizing word automata.

Two-way alternating tree automata (2ATA)

$A = (Q, \Sigma, I, \delta)$, where

$I \subseteq Q$ are "initial" states and

$\delta(s, q) \in B^+(Q \times \{-1, 0, 1, \dots, \text{arity}(s)\})$

$B^+(X)$ is the set of positive Boolean formulas over propositions X .

Example

$$\delta(h, q) = (q', 1) \wedge (q'', 1) \vee (q''', 2)$$

$$\delta(a, q') = \text{true}$$

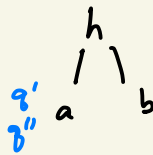
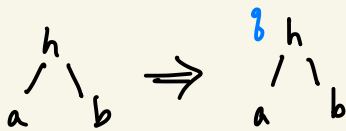
$$\delta(a, q'') = \text{true}$$

$$\delta(b, q''') = \text{false}$$

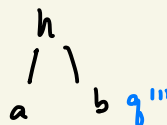
Means that in state q , reading symbol h , the automaton is allowed to **either** transition to the 1st child in state q' **and** to the 1st child in state q'' **or** to the 2nd child in state q''' .

Suppose $q \in I \subseteq Q$.

input is $h(a, b)$



Good run



X

Bad run

Example

$$\delta(s, q) = (q', 0) \vee (q'', -1)$$

move up on
input tree

$$\delta(s, q') = (q'', 1) \wedge (q''', -1)$$

$$\delta(f, q'') = \text{false}$$

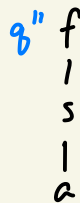
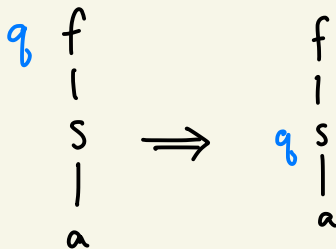
$$\delta(f, q''') = \text{true}$$

$$\delta(f, q) = (q, 1)$$

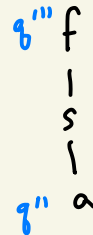
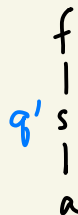
$$\delta(a, q'') = \text{true}$$

Suppose $q \in I \subseteq Q$.

input is $f(s(a))$



Bad run



Good run

Run of ZATA $A = (Q, \Sigma, I, \delta)$ on $t = (D(t), \lambda)$ is a $(Q \times D(t))$ -labeled tree $r_t = (D(r_t), \lambda_r)$ satisfying:

$$1) \lambda_r(\varepsilon) = (q_i, \varepsilon) \text{ for some } q_i \in I$$

$$2) \forall p \in D(r_t) \text{ with } \lambda_r(p) = (q, u)$$

have $M \models \delta(\lambda(u), q)$, where

$$M := \{ (q', d) : p' \in \text{Succ}(p), (q', u') = \lambda_r(p'), \\ d = \text{dir}(u, u') \neq \perp \},$$

$$\text{dir}(u, u') = \begin{cases} j & \text{if } u' = u \cdot j, j \in \{-1, 0, 1, \dots\} \\ \perp & \text{else} \end{cases},$$

$$\begin{aligned} \text{and } x \cdot 0 &= x & \forall x \in D(t) \\ x_i \cdot (-1) &= x & \forall x \in D(t), i > 0 \\ x \cdot i &= x_i & \forall x \in D(t), i > 0 \end{aligned}$$

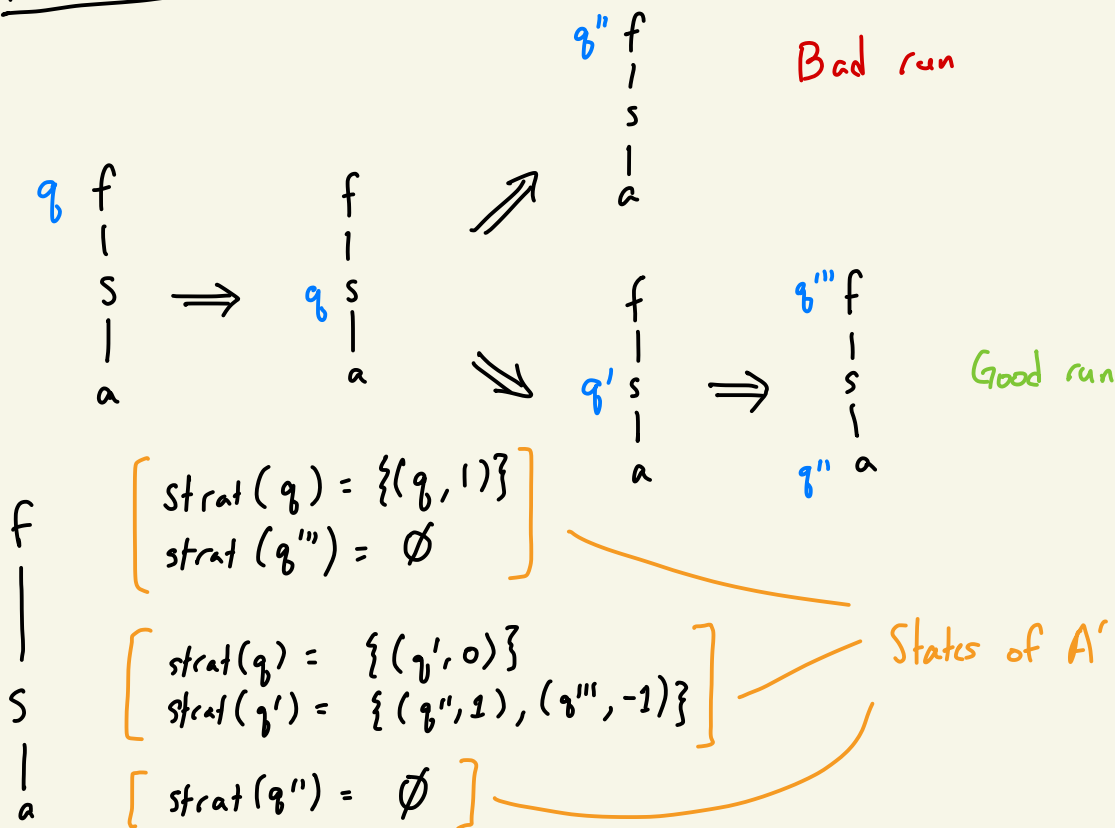
The language of a ZATA $A = (Q, \Sigma, I, \delta)$ is $L(A) := \{t \in T_\Sigma : A \text{ has a finite run on } t\}$.

Theorem: Let A be a ZATA.

There is a (bottom-up) tree automaton

A' with $L(A') = L(A)$ and A' can be computed in deterministic exponential time given A .

Proof idea: A' checks existence of a strategy for ZATA.



Proof: Fix $A = (Q, \Sigma, \mathcal{I}, \delta)$. We define bottom-up

$A' = (Q', \Sigma, F, \delta')$ with

$Q' = \text{Strategy}$ (summaries of how A could behave on specific tree vertices)

$$\left[\begin{array}{l} \text{Dir} := \{-1\} \cup \mathbb{N}, \text{Val} := \mathcal{P}^{\text{fin}}(Q \times \text{Dir}) \\ \text{Strategy} := [Q \rightarrow \text{Val}] \\ v \in \text{Val}, \text{Dirs}(v) := \{d \in \text{Dir} : (q, d) \in v\} \end{array} \right]$$

$$F = \{s \in \text{Strategy} : \text{dom}(s) \cap \mathcal{I} \neq \emptyset\}$$

$$\delta'_0(a) = \{s \in \text{Strategy} : \forall q \in \text{dom}(s). s(q) \models \delta_0(a, q)\}$$

$$\delta'_K(f, (s_1, \dots, s_K)) = \{s \in \text{Strategy} :$$

$$\begin{aligned} & \forall q \in \text{dom}(s). \forall i \in [K]. \{q' : (q', i) \in s(q)\} \subseteq \text{dom}(s_i) \\ & \wedge \forall i \in [K]. \forall q \in \text{dom}(s_i). \{q' : (q', -1) \in s_i(q)\} \subseteq \text{dom}(s) \} \end{aligned}$$

Claim: $L(A') = L(A)$

Finally we can complete the proof of decidability for $FOL(K)$ - separation.

Let's now define a ZATA

A_M for a positive example $M \in P$, s.t.

$$L(A_M) = \{ \text{ast}(\varphi) : M \models \varphi \}.$$

Use alphabet Σ_{simple} for $\tau_{\text{simple}} = \{ \mathbb{R}^2 \}$.

$$V = \{x, y\}.$$

 all partial assignments
to variables x, y

$$\text{Let } \Gamma := [V \rightarrow D(M)].$$

$$A_M = (Q, \Sigma_{\text{simple}}, \mathcal{I}, \delta), \quad \text{where}$$

$$Q = \Gamma \times \{0, 1\}$$

$$\mathcal{I} = \{(\emptyset, 0)\} \quad \text{assignment that's undefined everywhere}$$

$$\delta(\exists x, (\gamma, 0)) = \bigvee_{a \in D(M)} ((\gamma[x \mapsto a], 0), 1)$$

$$\delta(\forall x, (\gamma, 0)) = \bigwedge_{a \in D(M)} ((\gamma[x \mapsto a], 0), 1)$$

$$\delta(\wedge, (\gamma, 0)) = ((\gamma, 0), 1) \wedge ((\gamma, 0), z)$$

$$\delta(\vee, (\gamma, 0)) = ((\gamma, 0), 1) \vee ((\gamma, 0), z)$$

$$\delta(\neg, (\gamma, b)) = ((\gamma, \neg b), 1)$$

$$\delta(R(x, y), (\gamma, 0)) = \begin{cases} \text{true} & \text{if } x, y \in \text{dom}(\gamma) \\ & \text{and } M, \gamma \models R(x, y) \\ \text{false} & \text{else} \end{cases}$$

$$\delta(\forall x, (\gamma, 1)) = \bigvee_{a \in D(M)} ((\gamma[x \mapsto a], 1), 1)$$

$$\delta(\exists x, (\gamma, 1)) = \bigwedge_{a \in D(M)} ((\gamma[x \mapsto a], 1), 1)$$

$$\delta(R(x, y), (\gamma, 1)) = \begin{cases} \text{true} & \text{if } x, y \in \text{dom}(\gamma) \\ & \text{and } M, \gamma \not\models R(x, y) \\ \text{false} & \text{else} \end{cases}$$

... A few analogous cases for $\exists y, \forall y, R(x, x)$, etc.

Claim $L(A_M) = \{ \text{ast}(\varphi) \in T_{\Sigma, \text{simple}} : M \models \varphi \}$.