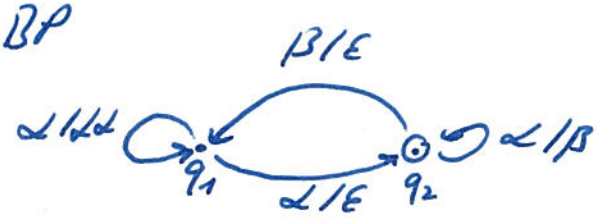
Example:

BP



Does $(q_1, \alpha\alpha\alpha)$ have an accepting run?

↳ Yes: $\big((q_1, \alpha\alpha\alpha) \to (q_2, \alpha\alpha) \to (q_2, \beta\alpha)$

$\to (q_1, \alpha) \to (q_1, \alpha\alpha) \to (q_1, \alpha\alpha\alpha)\big)^\omega$

Check this algorithmically.

Note:

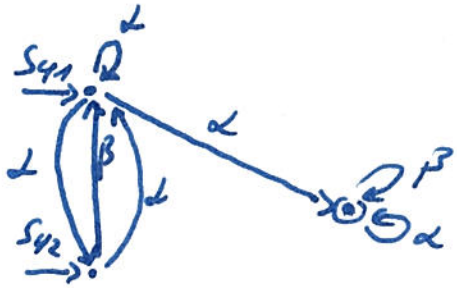
(q_1, α) seems to give rise to an accepting sequence

$$(q_1, \alpha) \to^+ (q_2, \alpha\alpha) \to^* (q_1, \alpha)$$
$$\| \qquad\qquad \| \qquad\qquad \|$$
$$(q, \gamma) \qquad (q_F, u) \qquad (q, \gamma \cdot w)$$

Check (by algorithm) that (2') holds for (q_1, α).

Determine

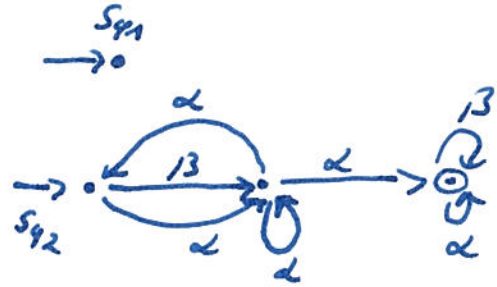
$pre^*(\{q_1\} \times \alpha \cdot \overline{T}^*)$:



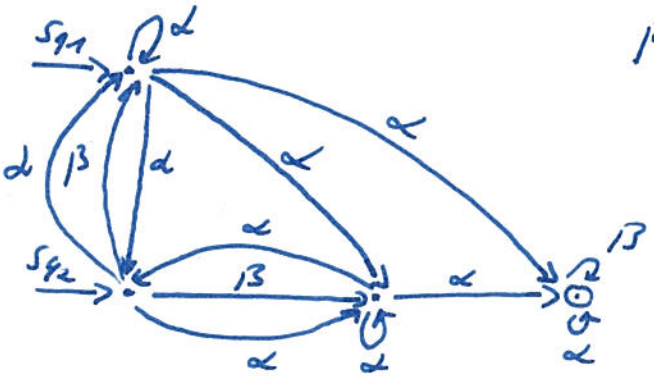
Intersect with $Q_F \times T^* = \{q_2\} \times T^*$:

BP-NFA for

$(\{q_2\} \times T^*) \cap pre^*(\{q_1\} \times \alpha \cdot T^*)$

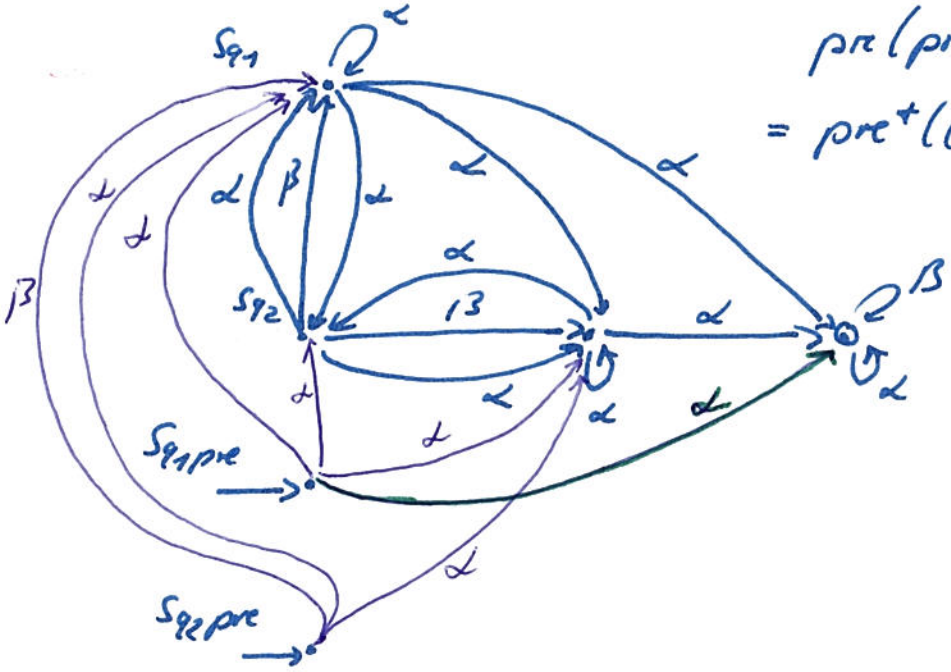


Determine pre^*:



$$pre^*((Q_F \times T^*) \cap pre^*(\{q_1\} \times \alpha . T^*))$$

Determine another single pre:



$$pre(pre^*((Q_F \times T^*) \cap pre^*(\{q_1\} \times \alpha . T^*)))$$
$$= pre^+((Q_F \times T^*) \cap pre^*(\{q_1\} \times \alpha . T^*))$$

Check:

$$(q_1, \alpha) \in pre^+((Q_F \times T^*) \cap pre^*(\{q_1\} \times \alpha . (\alpha+\beta)^*))$$
$$= pre^+((\{q_2\} \times (\alpha+\beta)^*) \cap pre^*(\{q_1\} \times \alpha . (\alpha+\beta)^*)).$$

This holds by $\xrightarrow{\alpha}$.

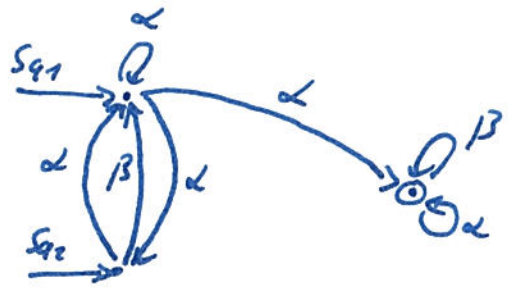
So configurations in $\{q_2\} \times \alpha . (\alpha+\beta)^*$ start an accepting run.

Construct (1'):

$$pre^*(\{q_1\} \times \alpha . T^*) = pre^*(\{q_1\} \times \alpha . (\alpha+\beta)^*).$$

Reuse from above

$\mathcal{A}$



Decide already here:

Does $(q_1, \alpha\alpha\alpha)$ have an accepting run?

Yes, because $(q_1, \alpha\alpha\alpha) \in L(\mathcal{A}) = pre^*(\{q_1\} \times \alpha \cdot (\alpha+\beta)^*)$.

Overview: further results on infinite words

MSO:

↳ Syntax of WMSO

↳ But interpreted over infinite words

⇒ Second-order quantifiers may range over infinite sets

$$\exists X: (\exists x: first(x) \wedge X(x)) \wedge (\forall x: X(x) \rightarrow \exists y: x < y \wedge X(y))$$

satisfiable in MSO.

Main result:

Satisfiability is decidable in MSO.

Proof:

↳ Construct NBA $\mathcal{A}_\varphi$ with

$$\mathcal{L}(\mathcal{A}_\varphi) = \{ w \in \Sigma^\omega \mid S_w \models \varphi \} = \mathcal{L}(\varphi).$$

⇒ Reuse techniques from Section 2.

↳ Check emptiness for $\mathcal{L}(\mathcal{A}_\varphi)$.

Rabin, Streett, and Muller automata

Recall: Büchi acceptance condition

infinite run

$r = q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} q_2 \xrightarrow{a_2} \dots$

satisfies

$\underbrace{\inf(r)}_{\text{states that are visited infinitely often.}} \cap Q_F \neq \emptyset.$

Rabin acceptance:

Final states are pairs $F = \{(G_1, F_1), \dots, (G_n, F_n)\}$

with $G_i, F_i \subseteq Q$ f.a. $1 \leq i \leq n$.

Run is accepting if

$\underbrace{\inf(r) \cap G_i \neq \emptyset}_{\text{states that should occur in the infinite}}$ and $\underbrace{\inf(r) \cap F_i = \emptyset}_{\text{states that are forbidden in the infinite}}$ for some $1 \leq i \leq k$

Streett acceptance:

- Dual of Rabin acceptance.
- Final states again pairs $F = \{(G_1, F_1), \dots, (G_n, F_n)\}$

with $G_i, F_i \subseteq Q$ f.a. $1 \leq i \leq n$.

Run is accepting if

$\inf(r) \cap G_i \neq \emptyset$ implies $\inf(r) \cap F_i \neq \emptyset$

for all $1 \leq i \leq n$

Muller acceptance:

Final states $F = \{Q_1, \dots, Q_n\}$ with $Q_i \subseteq Q$ f.a. $1 \leq i \leq n$

Run is accepting if $\inf(r) \in F$.

Note that Büchi acceptance is a special case of Rabin, Streett, Muller acceptance.

In fact, models define the same ω-languages:

$$\omega\text{-regular languages} = \text{NBA languages} \overset{\text{Rabin}}{=} \text{Streett} \overset{}{=} \text{Muller} \quad \text{languages.}$$

Determinisation with Safra:

Idea: Apply powerset construction to NBA.

↳ But not every NBA can be determinised.

↳ NBA $\rightsquigarrow$ DMA

deterministic Muller automaton

↳ States are trees with complex labelling

$\Rightarrow$ We use the Safraless algebraic approach, instead.

Algorithmic problems:

- Emptiness of NBAs when they are given as composition $Sys_1 \| \dots \| Sys_n$.
- Emptiness of Rabin / Streett / Muller automata.

6 Finite trees

Words = structures with positions and
one successor predicate $suc(x,y)$

Trees = structures with several successor predicates $suc_L(x,y)$, $suc_R(x,y)$

Important in Computer Science:

- Parse trees in CF grammars
- Abstract data types
- XML documents

Here: automata on trees

We already saw:

(word) languages = sets of finite words

ω-languages = sets of infinite words

Now: tree languages = sets of trees

Application:

Querying XML documents

Underlying problem:

What are the sets of trees
recognised by finite (tree) automata?

8. Top-down vs. bottom-up tree automata

Definition (Finite trees):

A finite tree is a subset $T \subseteq \mathbb{N}^*$
that satisfies the following closure properties:

- If $w.n \in T$ with $w \in \mathbb{N}^*$ and $n \in \mathbb{N}$ then $w \in T$.
// If a node is part of a tree, so is its father
- For $n > 0$ and $w.n \in T$ we have $w.(n-1) \in T$
// Children are consecutively labelled

Intuitively: node = path from the root to it.

Example: