Introduction to Model-Checking

Moritz Müller

Spring/Summer 2021

Finite automata

A nondeterministic finite automaton NFA is a tuple $\mathbb{A} = (S, \Sigma, s_0, \Delta, F)$:

- S is a set of states
- $s_0 \in S$ is the initial state
- Σ is a finite nonempty set, called alphabet; elements are letters
- $\Delta \subseteq S \times \Sigma \times S$ transition relation
- $F \subseteq S$ set of final states

A deterministic finite automaton DFA is an NFA with $\Delta : S \times \Sigma \to S$

Finite automata

A nondeterministic finite automaton NFA is a tuple $\mathbb{A} = (S, \Sigma, s_0, \Delta, F)$:

- S is a set of states
- $s_0 \in S$ is the initial state
- Σ is a finite nonempty set, called alphabet; elements are letters
- $\Delta \subseteq S \times \Sigma \times S$ transition relation
- $F \subseteq S$ set of final states

A deterministic finite automaton DFA is an NFA with $\Delta : S \times \Sigma \to S$

Executions $s_0 \; a_0 \; s_1 \; a_1 \; s_2 \cdots$ defined as before

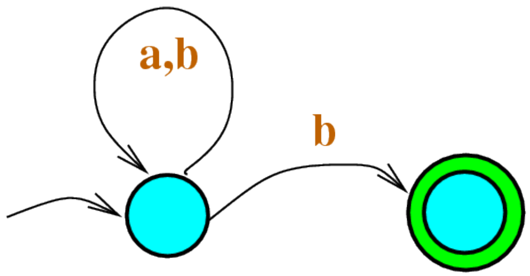
$\mathbb{A}$ accepts $a_0 \cdots a_{n-1}$ if there is such an execution with $s_n \in F$

$L(\mathbb{A}) := \{w \in \Sigma^+ \mid \mathbb{A} \text{ accepts } w\}$: sets of this form are regular languages

$\mathbb{A}$ and $\mathbb{B}$ are equivalent iff $L(\mathbb{A}) = L(\mathbb{B})$

Finite automata

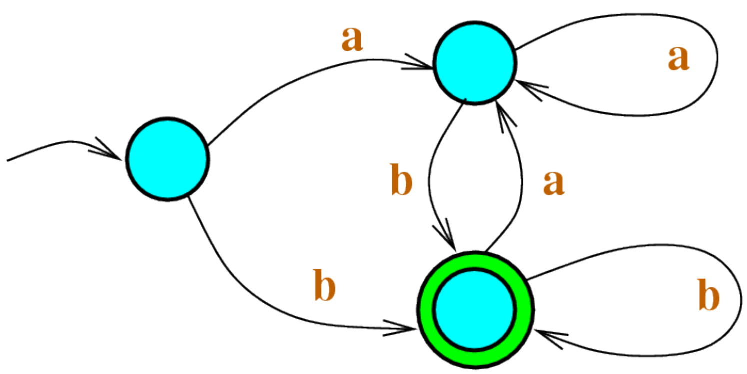
Examples



$\Sigma = \{a,b\}$

$L = (a+b)^*b$

NFA



DFA

Determinization

Proposition

Every NFA is equivalent to a DFA.

Proof Given an NFA $\mathbb{A} = (S, \Sigma, s_0, \Delta, F)$. Define $\mathbb{A}' = (S', \Sigma, s'_0, \Delta', F')$ by

- $S' := P(S)$
- $s'_0 := \{s_0\}$
- $\Delta' :=$ set of (X, a, Y) with $X \subseteq S, a \in \Sigma$ and

$$Y = \{s' \in S \mid (s, a, s') \in \Delta, s \in X\}$$

- $F' := \{X \subseteq S \mid X \cap F \neq \emptyset\}$

Then $\mathbb{A}'$ is a DFA with $L(\mathbb{A}) = L(\mathbb{A}')$. **qed**

Remark $(S', \Sigma, s'_0, \Delta', S' \setminus F')$ accepts $\Sigma^+ \setminus L(\mathbb{A})$.

Regular languages

Exercise

Regular languages are closed under Boolean operations and projections.

Regular languages

Exercise

Regular languages are closed under Boolean operations and projections.

More specifically:

(a) For every NFA $\mathbb{A}$ with k states there is a DFA $\mathbb{B}$ with 2^k states and $L(\mathbb{B}) = \Sigma^+ \setminus L(\mathbb{A})$

Regular languages

Exercise

Regular languages are closed under Boolean operations and projections.

More specifically:

(a) For every NFA $\mathbb{A}$ with k states there is a DFA $\mathbb{B}$ with 2^k states and $L(\mathbb{B}) = \Sigma^+ \setminus L(\mathbb{A})$

(b) For all NFAs $\mathbb{A}, \mathbb{A}'$ with k, k' states resp., there is an NFA $\mathbb{B}$ with $k + k' + 1$ states and $L(\mathbb{B}) = L(\mathbb{A}) \cup L(\mathbb{A}')$

Regular languages

Exercise

Regular languages are closed under Boolean operations and projections.

More specifically:

(a) For every NFA $\mathbb{A}$ with k states there is a DFA $\mathbb{B}$ with 2^k states and $L(\mathbb{B}) = \Sigma^+ \setminus L(\mathbb{A})$

(b) For all NFAs $\mathbb{A}, \mathbb{A}'$ with k, k' states resp., there is an NFA $\mathbb{B}$ with $k + k' + 1$ states and $L(\mathbb{B}) = L(\mathbb{A}) \cup L(\mathbb{A}')$

(c) For every NFA $\mathbb{A}$ over alphabet $\Sigma \times \Sigma'$ with k states, there is an NFA $\mathbb{B}$ with k states and

$$L(\mathbb{B}) = \Big\{ a_0 \cdots a_{n-1} \in \Sigma \mid n > 0, \exists b_0, \ldots b_{n-1} \in \Sigma' : (a_0, b_0) \cdots (a_{n-1}, b_{n-1}) \in L(\mathbb{A}) \Big\}$$

Regular languages

Exercise

Regular languages are closed under Boolean operations and projections.

More specifically:

(a) For every NFA $\mathbb{A}$ with k states there is a DFA $\mathbb{B}$ with 2^k states and $L(\mathbb{B}) = \Sigma^+ \setminus L(\mathbb{A})$

(b) For all NFAs $\mathbb{A}, \mathbb{A}'$ with k, k' states resp., there is an NFA $\mathbb{B}$ with $k + k' + 1$ states and $L(\mathbb{B}) = L(\mathbb{A}) \cup L(\mathbb{A}')$

(c) For every NFA $\mathbb{A}$ over alphabet $\Sigma \times \Sigma'$ with k states, there is an NFA $\mathbb{B}$ with k states and

$$L(\mathbb{B}) = \Big\{ a_0 \cdots a_{n-1} \in \Sigma \mid n > 0, \exists b_0, \ldots b_{n-1} \in \Sigma' : (a_0, b_0) \cdots (a_{n-1}, b_{n-1}) \in L(\mathbb{A}) \Big\}$$

(d) For every alphabet Σ' and NFA $\mathbb{A}$ over alphabet Σ with k states, there is an NFA $\mathbb{B}$ over $\Sigma \times \Sigma'$ with k states and

$$L(\mathbb{B}) = \Big\{ (a_0, b_0) \cdots (a_{n-1}, b_{n-1}) \in (\Sigma \times \Sigma')^+ \mid n > 0, a_0 \cdots a_{n-1} \in L(\mathbb{A}) \Big\}$$

Words as structures

View a word $w = a_0 \cdots a_{n-1} \in \Sigma^+$ as a structure $\mathcal{S}(w)$:

- vocabulary: $\{\leq\} \cup \{P_a \mid a \in \Sigma\}$:
- universe: $[n] = \{0, \ldots, n-1\}$
- $P_a^{\mathcal{S}(w)} := \{i \in [n] \mid a_i = a\}$
- $\leq^{\mathcal{S}(w)}$:= the natural $\leq$.

Words as structures

View a word $w = a_0 \cdots a_{n-1} \in \Sigma^+$ as a structure $\mathcal{S}(w)$:

- vocabulary: $\{\leq\} \cup \{P_a \mid a \in \Sigma\}$:
- universe: $[n] = \{0, \ldots, n-1\}$
- $P_a^{\mathcal{S}(w)} := \{i \in [n] \mid a_i = a\}$
- $\leq^{\mathcal{S}(w)}$:= the natural $\leq$.

An MSO-sentence φ defines

$$L(\varphi) = \{w \in \Sigma^+ \mid \mathcal{S}(w) \models \varphi\}$$

Büchi's Theorem 1960

Exactly the regular languages are MSO-definable.

Proof of Büchi's Theorem

Let $\mathbb{A}$ be an NFA, say, with $S = [k]$ and $s_0 = 0$.

Want $\varphi_{\mathbb{A}} \in$ MSO such that $L(\mathbb{A}) = L(\varphi_{\mathbb{A}})$.

Proof of Büchi's Theorem

Let $\mathbb{A}$ be an NFA, say, with $S = [k]$ and $s_0 = 0$.

Want $\varphi_{\mathbb{A}} \in$ MSO such that $L(\mathbb{A}) = L(\varphi_{\mathbb{A}})$.

$$\varphi_{\mathbb{A}} := \exists X_0 \cdots \exists X_{k-1} \Big(\textit{Part} \wedge \textit{Init} \wedge \textit{Trans} \wedge \textit{Acc} \Big)$$

Intuition: $X_i(x)$ means "$\mathbb{A}$ is in state i when reading the letter in position x".

Proof of Büchi's Theorem

Let $\mathbb{A}$ be an NFA, say, with $S = [k]$ and $s_0 = 0$.

Want $\varphi_{\mathbb{A}} \in$ MSO such that $L(\mathbb{A}) = L(\varphi_{\mathbb{A}})$.

$$\varphi_{\mathbb{A}} := \exists X_0 \cdots \exists X_{k-1} \Big(\mathit{Part} \wedge \mathit{Init} \wedge \mathit{Trans} \wedge \mathit{Acc} \Big)$$

Intuition: $X_i(x)$ means "$\mathbb{A}$ is in state i when reading the letter in position x".

- *Part* expresses that the X_i form a partition:

$$\forall x \Big(\bigvee_{i<k} X_i(x) \;\; \wedge \bigwedge_{i<j<k} (\neg X_i(x) \vee \neg X_j(x)) \Big).$$

- *Init* expresses that the computation starts in s_0:

$$\forall x (\forall z \; x \leq z \to X_0(x))$$

Proof of Büchi's Theorem

Let $\mathbb{A}$ be an NFA, say, with $S = [k]$ and $s_0 = 0$.

Want $\varphi_{\mathbb{A}} \in$ MSO such that $L(\mathbb{A}) = L(\varphi_{\mathbb{A}})$.

$$\varphi_{\mathbb{A}} := \exists X_0 \cdots \exists X_{k-1} \Big(\mathit{Part} \wedge \mathit{Init} \wedge \mathit{Trans} \wedge \mathit{Acc} \Big)$$

Intuition: $X_i(x)$ means "$\mathbb{A}$ is in state i when reading the letter in position x".

- *Trans* expresses that successive states accord to Δ:

$$\forall x \forall y \Big(x \leq y \wedge \neg x = y \wedge \forall z (z \leq x \vee y \leq z) \to \bigvee_{(i,a,j) \in \Delta} (X_i(x) \wedge P_a(x) \wedge X_j(y)) \Big)$$

- *Acc* expresses that the computation accepts:

$$\forall x \Big(\forall z \;\; z \leq x \to \bigvee_{\substack{(i,a,j) \in \Delta \\ j \in F}} (X_i(x) \wedge P_a(x)) \Big)$$

Proof of Büchi's Theorem

Let φ be an MSO-sentence in the vocabulary $\{\leq\} \cup \{P_a \mid a \in \Sigma\}$.

Want NFA $\mathbb{A}_\varphi$ over Σ such that $L(\varphi) = L(\mathbb{A}_\varphi)$.

Proof of Büchi's Theorem

Let φ be an MSO-sentence in the vocabulary $\{\leq\} \cup \{P_a \mid a \in \Sigma\}$.

Want NFA $\mathbb{A}_\varphi$ over Σ such that $L(\varphi) = L(\mathbb{A}_\varphi)$.

First step: massaging φ

$$\mathit{Sing}(X) := \exists x\big(X(x) \wedge \forall y(X(y) \to x = y)\big)$$

$$\mathit{Before}(X, Y) := \forall x, y\big(X(x) \wedge Y(y) \to x \leq y)\big)$$

$$\mathit{Letter}_a(X) := \forall x(X(x) \to P_a(x)) \qquad \text{for } a \in \Sigma.$$

Proof of Büchi's Theorem

Let φ be an MSO-sentence in the vocabulary $\{\leq\} \cup \{P_a \mid a \in \Sigma\}$.

Want NFA $\mathbb{A}_\varphi$ over Σ such that $L(\varphi) = L(\mathbb{A}_\varphi)$.

First step: massaging φ

$$\mathit{Sing}(X) := \exists x\big(X(x) \wedge \forall y(X(y) \to x = y)\big)$$

$$\mathit{Before}(X, Y) := \forall x, y\big(X(x) \wedge Y(y) \to x \leq y)\big)$$

$$\mathit{Letter}_a(X) := \forall x(X(x) \to P_a(x)) \qquad \text{for } a \in \Sigma.$$

An MSO-formula is ready for translation if it is obtained from the above formulas by means of $\neg, \vee, \exists Z$.

Claim: For every MSO-sentence φ there is an MSO-sentence φ^* that is ready for translation and such that

$$L(\varphi) = L(\varphi^*).$$

Proof of Büchi's Theorem

Proof of Claim: for $x, y, \ldots$ let $X, Y, \ldots$ be new set variables.

Define $\varphi(x, y, \ldots, \bar{Z}) \mapsto \varphi^*(X, Y, \ldots, \bar{Z})$ such that:

for all words $w \in \Sigma^+$, say of length n, we have

$$\mathcal{S}(w) \models \varphi(i, j, \ldots, \bar{A}) \quad \Longleftrightarrow \quad \mathcal{S}(w) \models \varphi^*(\{i\}, \{j\}, \ldots, \bar{A})$$

for all $i, j, \ldots \in [n]$ and all tuples $\bar{A}$ of subsets of $[n]$.

Proof of Büchi's Theorem

Proof of Claim: for $x, y, \ldots$ let $X, Y, \ldots$ be new set variables.

Define $\varphi(x, y, \ldots, \bar{Z}) \mapsto \varphi^*(X, Y, \ldots, \bar{Z})$ such that:

for all words $w \in \Sigma^+$, say of length n, we have

$$\mathcal{S}(w) \models \varphi(i, j, \ldots, \bar{A}) \iff \mathcal{S}(w) \models \varphi^*(\{i\}, \{j\}, \ldots, \bar{A})$$

for all $i, j, \ldots \in [n]$ and all tuples $\bar{A}$ of subsets of $[n]$.

$(x \leq y)^* := \mathit{Before}(X, Y)$

$(P_a(x))^* := \mathit{Letter}_a(X)$

$(\varphi \vee \psi)^* := \varphi^* \vee \psi^*$

$(\neg\varphi)^* := \neg\varphi^*$

$(\exists Z \varphi)^* := \exists Z \varphi^*$

$(\exists x \varphi)^* := \exists X(\mathit{Sing}(X) \wedge \varphi^*)$

The claim is proved.

Proof of Büchi's Theorem

Let $\varphi \in$ MSO. Want NFA $\mathbb{A}_\varphi$ such that $L(\varphi) = L(\mathbb{A}_\varphi)$.

Second step: translation

An MSO-formula $\varphi(Z_0, Z_1)$ defines

$$L\big(\varphi(Z_0, Z_1)\big) := \Big\{ w \in (\Sigma \times \{0,1\} \times \{0,1\})^+ \mid w \text{ satisfies } \varphi(Z_0, Z_1) \Big\}$$

Proof of Büchi's Theorem

Let $\varphi \in$ MSO. Want NFA $\mathbb{A}_\varphi$ such that $L(\varphi) = L(\mathbb{A}_\varphi)$.

Second step: translation

An MSO-formula $\varphi(Z_0, Z_1)$ defines

$$L\big(\varphi(Z_0, Z_1)\big) := \Big\{ w \in (\Sigma \times \{0,1\} \times \{0,1\})^+ \mid w \text{ satisfies } \varphi(Z_0, Z_1) \Big\}$$

Write $w \in (\Sigma \times \{0,1\} \times \{0,1\})^n$ as

$$w = (a_0, b_0^0, b_1^0) \cdots (a_{n-1}, b_0^{n-1}, b_1^{n-1})$$

w satisfies $\varphi(Z_0, Z_1)$ if $\mathcal{S}(a_0 \cdots a_{n-1}) \models \varphi(A_0, A_1)$,

where

$$\begin{aligned} A_0 &:= \big\{ i \in [n] \mid b_0^i = 1 \big\} \\ A_1 &:= \big\{ i \in [n] \mid b_1^i = 1 \big\} \end{aligned}$$

	a_0	a_1	a_2	a_3	a_4
$A_0 = \{1,3\}$	0	1	0	1	0
$A_1 = \{0,3\}$	1	0	0	1	0

Proof of Büchi's Theorem

Let φ be ready for translation. Write

$$\varphi = \varphi(\bar{Z})$$

where $\bar{Z}$ subsumes all (bound and free) set variables in φ.

Define $\mathbb{B}_{\varphi(\bar{Z})}$ such that $L(\mathbb{B}_{\varphi(\bar{Z})}) = L(\varphi(\bar{Z}))$:

Proof of Büchi's Theorem

Let φ be ready for translation. Write

$$\varphi = \varphi(\bar{Z})$$

where $\bar{Z}$ subsumes all (bound and free) set variables in φ.

Define $\mathbb{B}_{\varphi(\bar{Z})}$ such that $L(\mathbb{B}_{\varphi(\bar{Z})}) = L(\varphi(\bar{Z}))$:

- φ is $\mathit{Sing}(Z_i), \mathit{Letter}_a(Z_i), \mathit{Before}(Z_i, Z_j)$: Exercise!
- if φ is $(\psi \vee \chi)$, use closure under union Ex-(b).

  i.e., given $\mathbb{B}_{\psi(\bar{Z})}, \mathbb{B}_{\chi(\bar{Z})}$, choose $\mathbb{B}_{\varphi(\bar{Z})}$ such that

  $$L(\mathbb{B}_{\varphi(\bar{Z})}) = L(\mathbb{B}_{\psi(\bar{Z})}) \cup L(\mathbb{B}_{\chi(\bar{Z})}).$$

- if φ is $\neg\psi$, use closure under complementation Ex-(a).
- if φ is $\exists Z_i \psi$, use closure under projection Ex-(c) and padding Ex-(d).

Proof of Büchi's Theorem

Final move

given a MSO sentence φ,

compute φ^* ready for translation as described,

construct $\mathbb{B}_{\varphi^*}$ as described,

define $\mathbb{A}_\varphi$ over Σ from $\mathbb{B}_{\varphi^*}$ by projection.

Then $L(\mathbb{A}_\varphi) = L(\varphi)$. $\square$

Proof of Büchi's Theorem

Final move

given a MSO sentence φ,

compute φ^* ready for translation as described,

construct $\mathbb{B}_{\varphi^*}$ as described,

define $\mathbb{A}_\varphi$ over Σ from $\mathbb{B}_{\varphi^*}$ by projection.

Then $L(\mathbb{A}_\varphi) = L(\varphi)$. $\square$

Remark There described functions $\varphi \mapsto \mathbb{A}_\varphi$ and $\mathbb{A} \mapsto \varphi_{\mathbb{A}}$ are computable.

Corollaries of Büchi's Theorem: collapse of MSO over words

Let Σ be a finite alphabet.

Büchi's Theorem - effective version

There are computable functions

$$\varphi \mapsto \mathbb{A}_\varphi \text{ and } \mathbb{A} \mapsto \varphi_\mathbb{A}$$

from MSO$[\{\leq\} \cup \{P_a \mid a \in \Sigma\}]$-sentences to DFAs over Σ and back such that

$$L(\varphi) = L(\mathbb{A}_\varphi) \text{ and } L(\mathbb{A}) = L(\varphi_\mathbb{A}).$$

Corollaries of Büchi's Theorem: collapse of MSO over words

Let Σ be a finite alphabet.

Büchi's Theorem - effective version

There are computable functions

$$\varphi \mapsto \mathbb{A}_\varphi \text{ and } \mathbb{A} \mapsto \varphi_\mathbb{A}$$

from MSO$[\{\leq\} \cup \{P_a \mid a \in \Sigma\}]$-sentences to DFAs over Σ and back such that

$$L(\varphi) = L(\mathbb{A}_\varphi) \text{ and } L(\mathbb{A}) = L(\varphi_\mathbb{A}).$$

Corollary

There is a computable function that maps a given MSO$[\{\leq\} \cup \{P_a \mid a \in \Sigma\}]$-sentence ψ to an MSO$[\{\leq\} \cup \{P_a \mid a \in \Sigma\}]$-sentence φ of the form

$$\exists \bar{X} \;\; \varphi_0$$

where φ_0 is first-order such that for all $w \in \Sigma^+$:

$$\mathcal{S}(w) \models \psi \iff \mathcal{S}(w) \models \varphi.$$

Proof Set $\varphi := \varphi_{\mathbb{A}_\psi}$ as in the proof of Büchi's theorem. $\square$

Corollaries of Büchi's Theorem: model-checking MSO over words

Corollary The problem

Input: $w \in \Sigma^+$, MSO sentence φ.
Problem: $\mathcal{S}(w) \models \varphi$.

is decidable in time $O\big(f(|\varphi|) + |w|\big)$ for some function $f : \mathbb{N} \to \mathbb{N}$.

Corollaries of Büchi's Theorem: model-checking MSO over words

Corollary The problem

Input: $w \in \Sigma^+$, MSO sentence φ.
Problem: $\mathcal{S}(w) \models \varphi$.

is decidable in time $O\big(f(|\varphi|) + |w|\big)$ for some function $f : \mathbb{N} \to \mathbb{N}$.

Proof Given w, φ compute the DFA $\mathbb{A}_\varphi = (S, s_0, \Delta, F)$.

Check whether $\mathbb{A}_\varphi$ accepts $w = a_0 \cdots a_{n-1}$:

$s \leftarrow s_0$

$i \leftarrow 0$

while $i < n$ **do:**

$a \leftarrow a_i$

$s \leftarrow \Delta(s, a)$

$i \leftarrow i + 1$

if $s \in F$, **accept, else reject.**

We assume each line needs constant time. $\square$

Corollaries of Büchi's Theorem: MSO inexpressibility over words

Rabin, Scott 1959: Pumping Lemma Let L be regular.

There is $p \in \mathbb{N}$ such that every $w \in L$ with $|w| \geq p$ can be written

$$w = xyz$$

with $|xy| \leq p$ and y not empty such that for all $n \in \mathbb{N}$: $xy^n z \in L$.

Corollaries of Büchi's Theorem: MSO inexpressibility over words

Rabin, Scott 1959: Pumping Lemma Let L be regular.

There is $p \in \mathbb{N}$ such that every $w \in L$ with $|w| \geq p$ can be written

$$w = xyz$$

with $|xy| \leq p$ and y not empty such that for all $n \in \mathbb{N}$: $xy^n z \in L$.

Proof

Let $\mathbb{A} = (S, s_0, \Delta, F)$ be an NFA with $L(\mathbb{A}) = L$. Let $p := |S|$.

Let $w = a_0 \cdots a_{n-1} \in L$ with $n \geq p$ and let

$$s_0 \ a_0 \ s_1 \ a_1 \ s_2 \ a_2 \ \cdots s_{n-1} \ a_{n-1} \ s_n$$

be an execution with $s_n \in F$.

Corollaries of Büchi's Theorem: MSO inexpressibility over words

Rabin, Scott 1959: Pumping Lemma Let L be regular.

There is $p \in \mathbb{N}$ such that every $w \in L$ with $|w| \geq p$ can be written

$$w = xyz$$

with $|xy| \leq p$ and y not empty such that for all $n \in \mathbb{N}$: $xy^n z \in L$.

Proof

Let $\mathbb{A} = (S, s_0, \Delta, F)$ be an NFA with $L(\mathbb{A}) = L$. Let $p := |S|$.

Let $w = a_0 \cdots a_{n-1} \in L$ with $n \geq p$ and let

$$s_0 \ a_0 \ s_1 \ a_1 \ s_2 \ a_2 \ \cdots s_{n-1} \ a_{n-1} \ s_n$$

be an execution with $s_n \in F$. Choose $i < j \leq n$ with $s_i = s_j$. Set

$$x := a_0 \cdots a_{i-1}$$
$$y := a_i \cdots a_{j-1}$$
$$z := a_j \cdots a_{n-1}$$

Repeating $a_i \ s_{i+1} \ \cdots a_{j-1} \ s_j$ for n times is again an execution. $\square$

Corollaries of Büchi's Theorem: MSO inexpressibility over words

Rabin, Scott 1959: Pumping Lemma Let L be regular.

There is $p \in \mathbb{N}$ such that every $w \in L$ with $|w| \geq p$ can be written

$$w = xyz$$

with $|xy| \leq p$ and y not empty such that for all $n \in \mathbb{N}$: $xy^nz \in L$.

Example $\{a^k b^k \mid k > 0\}$ is not regular, hence not MSO-definable.

Corollaries of Büchi's Theorem: MSO inexpressibility over words

Rabin, Scott 1959: Pumping Lemma Let L be regular.

There is $p \in \mathbb{N}$ such that every $w \in L$ with $|w| \geq p$ can be written

$$w = xyz$$

with $|xy| \leq p$ and y not empty such that for all $n \in \mathbb{N}$: $xy^n z \in L$.

Example $\{a^k b^k \mid k > 0\}$ is not regular, hence not MSO-definable.

Example

View a word over $\Sigma = \{a, b\}$ as a tachograph recording:
a means "driving", b means "resting"

Law: "every driving time must be followed by an equally long time of resting."

Legal tachogaphs recordings:

$$L := \{b^m a^{i_1} b^{i_1} \cdots a^{i_n} b^{i_n} \mid n, m \in \mathbb{N}, i_1, \dots, i_n \in \mathbb{N}\}$$

Not MSO-definable (Exercise).

Corollaries of Büchi's Theorem: MSO inexpressibility over words

Rabin, Scott 1959: Pumping Lemma Let L be regular.

There is $p \in \mathbb{N}$ such that every $w \in L$ with $|w| \geq p$ can be written

$$w = xyz$$

with $|xy| \leq p$ and y not empty such that for all $n \in \mathbb{N}$: $xy^nz \in L$.

Example $\{a^k b^k \mid k > 0\}$ is not regular.

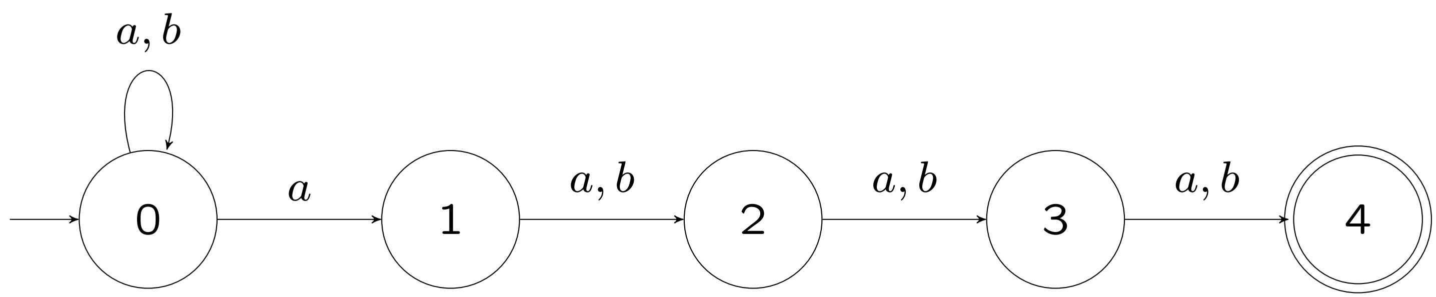
Exercise

There is no MSO$[\{\leq\} \cup \{P_a, P_b\}]$-formula $\varphi(x, y, z)$ such that for all $w \in \{a, b\}^+$ and all $i, j, k \in [|w|]$

$$i + j = k \iff \mathcal{S}(w) \models \varphi(i, j, k).$$

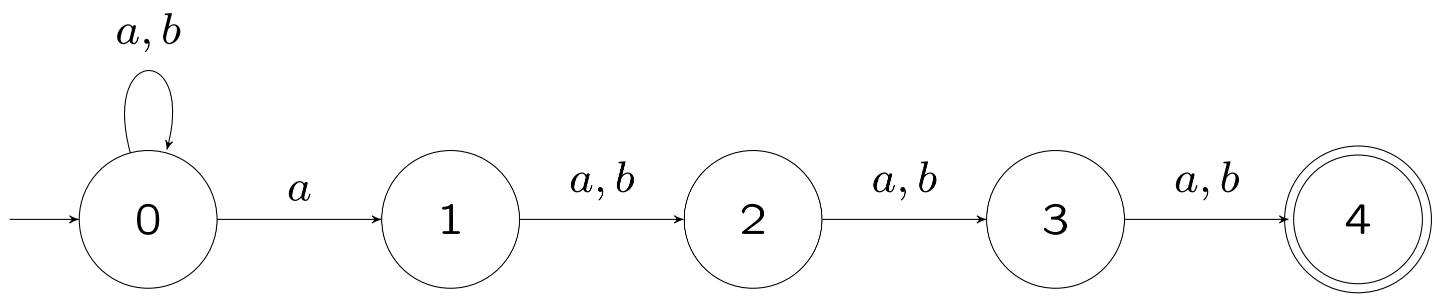
Lower bounds

NFA $\mathbb{A}$:



Lower bounds

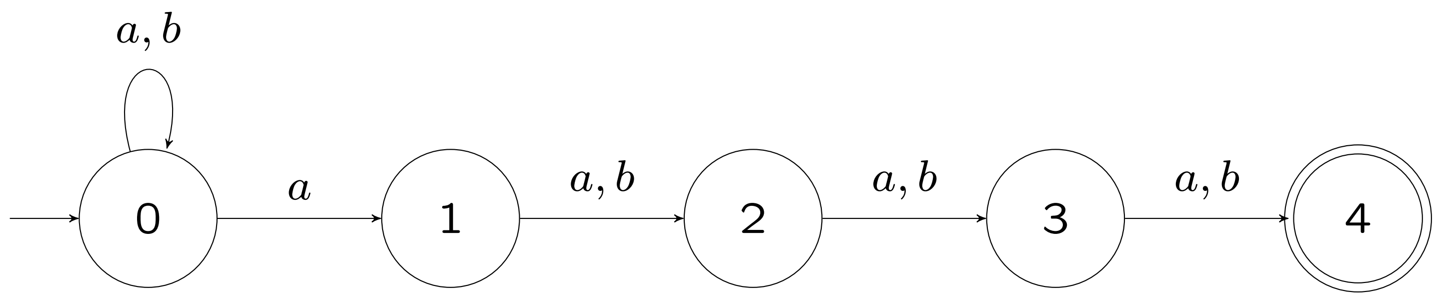
NFA $\mathbb{A}$:



$L(\mathbb{A}) = L_4 := \{w \in \{a, b\}^+ \mid \text{4th letter from right in } w \text{ is } a\}$

Lower bounds

NFA $\mathbb{A}$:



$L(\mathbb{A}) = L_4 := \{w \in \{a,b\}^+ \mid \text{4th letter from right in } w \text{ is } a\}$

Proposition Let $k \in \mathbb{N}_{>0}$. Every DFA $\mathbb{A}$ with $L(\mathbb{A}) = L_k$ has at least 2^k states.

Proof Assume $\mathbb{A}$ is a DFA with $< 2^k$ states.

There exists distinct $x = x_0 \cdots x_{k-1}, y = y_0 \cdots y_{k-1} \in \{a,b\}^k$ such that $\mathbb{A}$ on x, y reaches the same state.

Say, $x_i \neq y_i$. Then $\mathbb{A}$ accepts xb^{k-i} iff $\mathbb{A}$ accepts yb^{k-i}.

Exactly one is in L_k. Hence $L(\mathbb{A}) \neq L_k$. $\square$

Lower bounds

Corollary

The problem

Input: $w \in \Sigma^+$, MSO sentence φ.
Problem: $\mathcal{S}(w) \models \varphi$.

is decidable in time $O\big(f(|\varphi|) + |w|\big)$ for some computable $f : \mathbb{N} \to \mathbb{N}$.

Lower bounds

Frick, Grohe 2004

Assume P $\neq$ NP.

The problem

Input: $w \in \Sigma^+$, MSO sentence φ.
Problem: $\mathcal{S}(w) \models \varphi$.

is not decidable in time $O\big(f(|\varphi|) \cdot |w|^c\big)$ for any $c \in \mathbb{N}$ and elementary f.

Lower bounds

Frick, Grohe 2004

Assume P $\neq$ NP.

The problem

Input: $w \in \Sigma^+$, MSO sentence φ.
Problem: $\mathcal{S}(w) \models \varphi$.

is not decidable in time $O\big(f(|\varphi|) \cdot |w|^c\big)$ for any $c \in \mathbb{N}$ and elementary f.

$f : \mathbb{N} \to \mathbb{N}$ is elementary if there is $h \in \mathbb{N}$ such that for all $k \in \mathbb{N}$:

$$f(k) \leq 2^{2^{\cdot^{\cdot^{\cdot^{2^k}}}}} \quad (h\text{-fold exponential}).$$

Lower bounds

Frick, Grohe 2004

Assume FPT $\neq$ AW$[*]$.

The problem

Input: $w \in \Sigma^+$, FO sentence φ.
Problem: $\mathcal{S}(w) \models \varphi$.

is not decidable in time $O\big(f(|\varphi|) \cdot |w|^c\big)$ for any $c \in \mathbb{N}$ and elementary f.

$f : \mathbb{N} \to \mathbb{N}$ is elementary if there is $h \in \mathbb{N}$ such that for all $k \in \mathbb{N}$:

$$f(k) \leq 2^{2^{\cdot^{\cdot^{\cdot^{2^k}}}}} \quad (h\text{-fold exponential}).$$

ω-regular languages

A (non)deterministic Büchi automaton (NBA) DBA is an (NFA) DBA

$$\mathbb{A} = (S, s_0, \Delta, F).$$

$\mathbb{A}$ accepts an infinite word

$$\sigma = a_0\ a_1\ a_2 \cdots \in \Sigma^\omega$$

if there exists an execution

$$s_0\ a_0\ s_1\ a_1\ s_2\ a_2\ s_3 \cdots$$

such that $s_i \in F$ for infinitely many $i \in \mathbb{N}$.

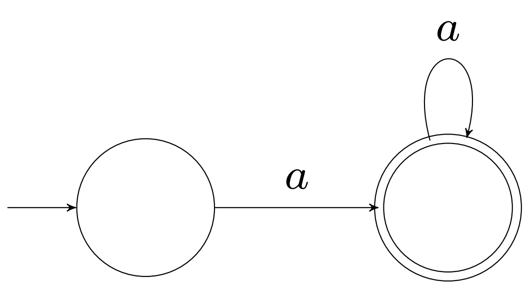
An ω-regular language is a subset of Σ^ω of the form

$$L_\omega(\mathbb{A}) := \{\sigma \in \Sigma^\omega \mid \mathbb{A} \text{ accepts } \sigma\}$$

for some NBA $\mathbb{A}$.

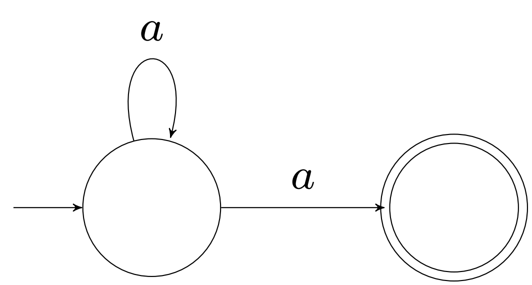
Examples

$\mathbb{A}$



$L(\mathbb{A}) = \{a\}^+$
$L_\omega(\mathbb{A}) = \{aaa\cdots\}$

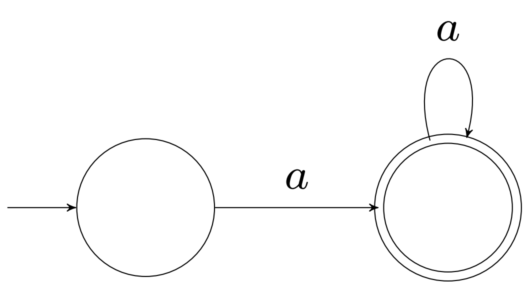
$\mathbb{B}$



$L(\mathbb{B}) = \{a\}^+$
$L_\omega(\mathbb{B}) = \emptyset$

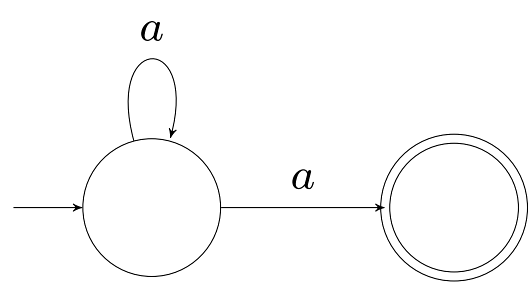
Examples

$\mathbb{A}$



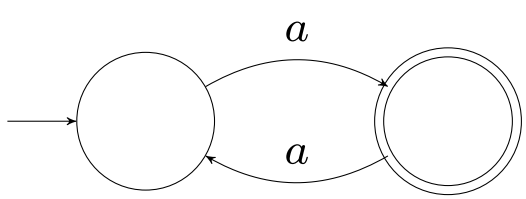
$L(\mathbb{A}) = \{a\}^+$
$L_\omega(\mathbb{A}) = \{aaa\cdots\}$

$\mathbb{B}$



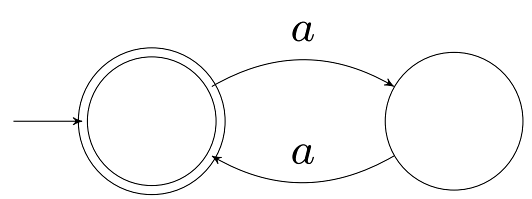
$L(\mathbb{B}) = \{a\}^+$
$L_\omega(\mathbb{B}) = \emptyset$

$\mathbb{A}$



$L(\mathbb{A}) = \{a^{2n+1} \mid n \in \mathbb{N}\}$
$L_\omega(\mathbb{A}) = \{aaa\cdots\}$

$\mathbb{B}$



$L(\mathbb{B}) = \{a^{2n} \mid n \in \mathbb{N}_{>0}\}$
$L_\omega(\mathbb{B}) = \{aaa\cdots\}$

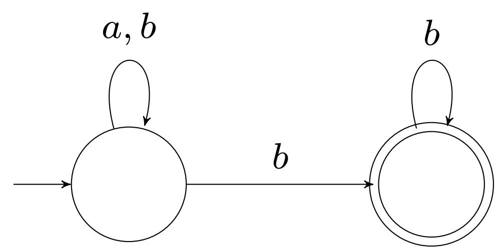
Determinization fails

Proposition

There is an ω-regular language L such that $L \neq L_\omega(\mathbb{A})$ for every DBA $\mathbb{A}$.

Proof Let $\Sigma = \{a, b\}$ and let L contain the words with finitely many a.

L is ω-regular:



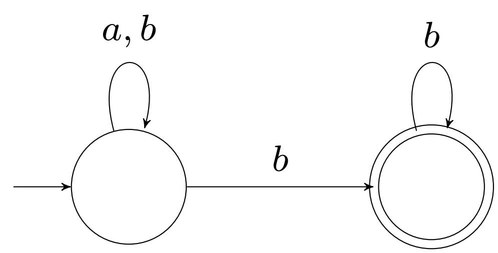
Determinization fails

Proposition

There is an ω-regular language L such that $L \neq L_\omega(\mathbb{A})$ for every DBA $\mathbb{A}$.

Proof Let $\Sigma = \{a, b\}$ and let L contain the words with finitely many a.

L is ω-regular:



Let $\mathbb{A}$ be a DBA and assume $L_\omega(\mathbb{A}) = L$.

Its accepting run on $b\ b\ b\ b \cdots$ visits a final state, say after reading b^{n_0}.

This run is continued to an accepting run of $b^{n_0}\ a\ b\ b\ b \cdots \in L$.

Choose n_1 such that $\mathbb{A}$ is in a final state after reading $b^{n_0} a b^{n_1}$.

Continue. Get accepting run on

$$b^{n_0}\ a\ b^{n_1}\ a\ b^{n_2}\ a \cdots$$

Outside L, contradiction. $\square$

Complementation

McNaughton 1966

The set of ω-regular languages is effectively closed under complementation:

> there is a computable function that maps an NBA $\mathbb{A}$ to an NBA $\mathbb{B}$ such that $\Sigma^\omega \setminus L_\omega(\mathbb{A}) = L_\omega(\mathbb{B})$.

Proof omitted. As before:

Corollary

The set of ω-regular languages is effectively closed under Boolean combinations and projections.

Complementation

McNaughton 1966

The set of ω-regular languages is effectively closed under complementation:

> there is a computable function that maps an NBA $\mathbb{A}$ to an NBA $\mathbb{B}$ such that $\Sigma^\omega \setminus L_\omega(\mathbb{A}) = L_\omega(\mathbb{B})$.

Proof omitted. As before:

Corollary

The set of ω-regular languages is effectively closed under Boolean combinations and projections.

Intersection

- A generalized NBA $\mathbb{A}$ is a tuple $(S, s_0, \Delta, \mathcal{F})$ like an NBA but with $\mathcal{F} \subseteq 2^S$.
- $\mathbb{A}$ accepts $a_0\ a_1 \cdots \in \Sigma^\omega$ iff there is an execution $s_0\ a_0\ s_1\ a_1 \cdots$ such that
  for all $F \in \mathcal{F}$ there are infinitely many $i \in \mathbb{N}$ with $s_i \in F$.

Complementation

McNaughton 1966

The set of ω-regular languages is effectively closed under complementation:

> there is a computable function that maps an NBA $\mathbb{A}$ to an NBA $\mathbb{B}$ such that $\Sigma^\omega \setminus L_\omega(\mathbb{A}) = L_\omega(\mathbb{B})$.

Proof omitted. As before:

Corollary

The set of ω-regular languages is effectively closed under Boolean combinations and projections.

Intersection

- A generalized NBA $\mathbb{A}$ is a tuple $(S, s_0, \Delta, \mathcal{F})$ like an NBA but with $\mathcal{F} \subseteq 2^S$.
- $\mathbb{A}$ accepts $a_0\ a_1 \cdots \in \Sigma^\omega$ iff there is an execution $s_0\ a_0\ s_1\ a_1 \cdots$ such that

  for all $F \in \mathcal{F}$ there are infinitely many $i \in \mathbb{N}$ with $s_i \in F$.

Exercise For every GNBA $\mathbb{A}$ there is an NBA $\mathbb{B}$ st $L_\omega(\mathbb{A}) = L_\omega(\mathbb{B})$.

Exercise For all GNBAs $\mathbb{A}, \mathbb{A}'$ there is a GNBA $\mathbb{B}$ st $L_\omega(\mathbb{A}) \cap L_\omega(\mathbb{A}') = L_\omega(\mathbb{B})$.

Büchi again

Let Σ be a finite alphabet. View $\sigma = a_0\ a_1 \cdots \in \Sigma^\omega$ as a structure $\mathcal{S}(\sigma)$:

- vocabulary: $\{\leq\} \cup \{P_a \mid a \in \Sigma\}$:
- universe: $\mathbb{N}$
- $P_a^{\mathcal{S}(\sigma)} := \{i \in \mathbb{N} \mid a_i = a\}$
- $\leq^{\mathcal{S}(\sigma)} :=$ the natural $\leq$.

An MSO-sentence φ defines $L_\omega(\varphi) = \{\sigma \in \Sigma^\omega \mid \mathcal{S}(\sigma) \models \varphi\}$

Büchi's theorem - ω-version

There are computable functions

$$\varphi \mapsto \mathbb{A}_\varphi \text{ and } \mathbb{A} \mapsto \varphi_\mathbb{A}$$

from MSO-sentences to NBAs and back such that

$$L_\omega(\varphi) = L_\omega(\mathbb{A}_\varphi) \text{ and } L_\omega(\mathbb{A}) = L_\omega(\varphi_\mathbb{A}).$$

Proof As before. Exercise Define $\mathbb{A} \mapsto \varphi_\mathbb{A}$. $\square$

Corollaries

Corollary The following problems are decidable.

Input: MSO$[\{\leq\} \cup \{P_a \mid a \in \Sigma\}]$-sentence φ.

Problem: is there a $\sigma \in \Sigma^\omega$ such that $\mathcal{S}(\sigma) \models \varphi$?

Input: MSO$[\{\leq\} \cup \{P_a \mid a \in \Sigma\}]$-sentences φ, ψ.

Problem: are φ and ψ equivalent in all structures $\mathcal{S}(\sigma)$ for $\sigma \in \Sigma^\omega$?

Proof Second follows from first.

First: compute $\mathbb{A}_\varphi$, check whether $L_\omega(\mathbb{A}_\varphi) = \emptyset$.

Equivalently: check whether there is a final state that is reachable from the initial state and lies on a cycle. $\square$

Linear time properties

Transition system $\mathbb{T}$ consists of:

S set of states

$I \subseteq S$ a set of initial states

Act set of states

$\to \subseteq S \times Act \times S$ transition relation

AP set of propositional variables

$L : S \to 2^{AP}$ labeling

Additional assumption: for all $s \in S$ there are $\alpha \in Act, s' \in S : s \xrightarrow{\alpha} s'$

The trace of an execution $s_0\ \alpha_0\ s_1\ \alpha_1\ s_2\ \alpha_2 \cdots$ is

$$L(s_0)\ L(s_1)\ L(s_2) \cdots \in \Sigma^\omega$$

where $\Sigma := 2^{AP}$.

Linear time property: subset $P \subseteq \Sigma^\omega$.

$\mathbb{T}$ satisfies P if every trace of (an execution of) $\mathbb{T}$ is in P.

Linear time properties

- closure $\quad \mathsf{cl}(P) := \{\sigma \in \Sigma^\omega \mid$ every finite prefix of σ is a prefix of some $\tau \in P\}$

  Exercise $\quad P \subseteq \mathsf{cl}(P), \quad \mathsf{cl}(\mathsf{cl}(P)) = \mathsf{cl}(P), \quad \mathsf{cl}(P \cup Q) = \mathsf{cl}(P) \cup \mathsf{cl}(Q)$

Linear time properties

- closure $\quad \mathsf{cl}(P) := \{\sigma \in \Sigma^\omega \mid$ every finite prefix of σ is a prefix of some $\tau \in P\}$

  Exercise $\quad P \subseteq \mathsf{cl}(P), \quad \mathsf{cl}(\mathsf{cl}(P)) = \mathsf{cl}(P), \quad \mathsf{cl}(P \cup Q) = \mathsf{cl}(P) \cup \mathsf{cl}(Q)$

- "something bad never happens": P safety property iff $\mathsf{cl}(P) = P$

  iff every $\sigma \notin P$ has a P-bad prefix (no element of P has this prefix)

Linear time properties

- closure $\mathsf{cl}(P) := \{\sigma \in \Sigma^\omega \mid \text{ every finite prefix of } \sigma \text{ is a prefix of some } \tau \in P\}$

  Exercise $P \subseteq \mathsf{cl}(P)$, $\mathsf{cl}(\mathsf{cl}(P)) = \mathsf{cl}(P)$, $\mathsf{cl}(P \cup Q) = \mathsf{cl}(P) \cup \mathsf{cl}(Q)$

- "something bad never happens": P safety property iff $\mathsf{cl}(P) = P$

  iff every $\sigma \notin P$ has a P-bad prefix (no element of P has this prefix)

  Exercise Two TSs have the same finite prefixes of traces
  iff satisfy the same safety properties
  (for finite TSs) iff trace-equivalent (have the same traces).

Linear time properties

- closure $\quad \mathsf{cl}(P) := \{\sigma \in \Sigma^\omega \mid$ every finite prefix of σ is a prefix of some $\tau \in P\}$

  Exercise $\quad P \subseteq \mathsf{cl}(P), \quad \mathsf{cl}(\mathsf{cl}(P)) = \mathsf{cl}(P), \quad \mathsf{cl}(P \cup Q) = \mathsf{cl}(P) \cup \mathsf{cl}(Q)$

- "something bad never happens": P safety property iff $\mathsf{cl}(P) = P$

  iff every $\sigma \notin P$ has a P-bad prefix (no element of P has this prefix)

  Exercise Two TSs have the same finite prefixes of traces
  iff satisfy the same safety properties
  (for finite TSs) iff trace-equivalent (have the same traces).

- "something good will happen": P liveness property iff $\mathsf{cl}(P) = \Sigma^\omega$

  iff every $w \in \Sigma^+$ is prefix of some $\sigma \in P$.

Linear time properties

- closure $\mathsf{cl}(P) := \big\{\sigma \in \Sigma^\omega \mid$ every finite prefix of σ is a prefix of some $\tau \in P\big\}$

  Exercise $P \subseteq \mathsf{cl}(P)$, $\mathsf{cl}(\mathsf{cl}(P)) = \mathsf{cl}(P)$, $\mathsf{cl}(P \cup Q) = \mathsf{cl}(P) \cup \mathsf{cl}(Q)$

- "something bad never happens": P safety property iff $\mathsf{cl}(P) = P$

  iff every $\sigma \notin P$ has a P-bad prefix (no element of P has this prefix)

  Exercise Two TSs have the same finite prefixes of traces
  iff satisfy the same safety properties
  (for finite TSs) iff trace-equivalent (have the same traces).

- "something good will happen": P liveness property iff $\mathsf{cl}(P) = \Sigma^\omega$

  iff every $w \in \Sigma^+$ is prefix of some $\sigma \in P$.

Remark

Every $P \subseteq \Sigma^\omega$ is the intersection of a safety and a liveness property, namely:

$$P = \mathsf{cl}(P) \cap (P \cup (\Sigma^\omega \setminus \mathsf{cl}(P)))$$

Indeed: $\mathsf{cl}\big(P \cup (\Sigma^\omega \setminus \mathsf{cl}(P))\big) = \mathsf{cl}(P) \cup \mathsf{cl}(\Sigma^\omega \setminus \mathsf{cl}(P)) = \Sigma^\omega$

Trace automaton

Proposition The set of traces of a transition system is ω-regular.

Proof Given $\mathbb{T} = (S, I, Act, \rightarrow, AP, L)$.

Define NBA $\mathbb{A}_{\mathbb{T}}$:

> states $S \cup \{s_0\}$ for a new $s_0 \notin S$
>
> initial state s_0
>
> alphabet $\Sigma := 2^{AP}$
>
> transition relation contains (s, a, s') iff
>
> > $a = L(s)$ and $s \stackrel{\textit{any}}{\rightarrow} s'$ (i.e., $s \stackrel{\alpha}{\rightarrow} s'$ for some $\alpha \in Act$), or,
> >
> > $s = s_0$ and $s'' \stackrel{\textit{any}}{\rightarrow} s'$ and $a = L(s'')$ for some $s'' \in I$.
>
> final states S

Then $L_\omega(\mathbb{A}_{\mathbb{T}}) =$ the set of traces of $\mathbb{T}$. $\square$

Model-checking MSO-definable linear time properties

Main Theorem The problem

Input: transition system $\mathbb{T}$, MSO sentence φ.

Problem: $\mathcal{S}(\sigma) \models \varphi$ for every trace σ of $\mathbb{T}$?

is decidable in time $O(f(|\varphi|) \cdot |\mathbb{T}|)$ for some computable $f : \mathbb{N} \to \mathbb{N}$.

Model-checking MSO-definable linear time properties

Main Theorem The problem

Input: transition system $\mathbb{T}$, MSO sentence φ.
Problem: $\mathcal{S}(\sigma) \models \varphi$ for every trace σ of $\mathbb{T}$?

is decidable in time $O(f(|\varphi|) \cdot |\mathbb{T}|)$ for some computable $f : \mathbb{N} \to \mathbb{N}$.

Proof Compute trace automaton $\mathbb{A}_{\mathbb{T}}$ of size $O(|\mathbb{T}|)$

Compute NBA $\mathbb{A}_{\neg\varphi}$ of Büchi's Theorem

Compute NBA $\mathbb{B}$ of size $O(|\mathbb{A}_{\neg\varphi}| \cdot |\mathbb{A}_{\mathbb{T}}|)$ (earlier Exercise) with

$$L_\omega(\mathbb{B}) = L_\omega(\mathbb{A}_{\mathbb{T}}) \cap L_\omega(\mathbb{A}_{\neg\varphi}).$$

Check whether $L_\omega(\mathbb{B}) = \emptyset$ (iff $L_\omega(\mathbb{A}_{\mathbb{T}}) \subseteq L_\omega(\varphi)$)

check no cycle contains a reachable final state
standard techniques do this in time $O(|\mathbb{B}|)$ $\square$

Linear temporal logic

LTL-formulas over a set of propositional variables AP generated by

$$\frac{}{p}\,p\in AP \qquad \frac{\varphi}{\neg\varphi} \qquad \frac{\varphi \quad \psi}{(\varphi \wedge \psi)} \qquad \frac{\varphi}{X\varphi} \qquad \frac{\varphi \quad \psi}{(\varphi U \psi)}$$

Linear temporal logic

LTL-formulas over a set of propositional variables AP generated by

$$\frac{}{p}p \in AP \qquad \frac{\varphi}{\neg\varphi} \qquad \frac{\varphi \quad \psi}{(\varphi \wedge \psi)} \qquad \frac{\varphi}{X\varphi} \qquad \frac{\varphi \quad \psi}{(\varphi U \psi)}$$

LTL semantics over $\sigma = a_0\ a_1\ \cdots \in \Sigma^\omega$ for $\Sigma := 2^{AP}$ and $i \in \mathbb{N}$:

$$\begin{array}{lcl}
\sigma, i \models p & \iff & p \in a_i \\
\sigma, i \models \neg\varphi & \iff & \sigma, i \not\models \varphi \\
\sigma, i \models (\varphi \wedge \psi) & \iff & \sigma, i \models \varphi \text{ and } \sigma, i \models \psi
\end{array}$$

Next

$$\sigma, i \models X\varphi \iff \sigma, i+1 \models \varphi$$

Until

$$\sigma, i \models (\varphi U \psi) \iff \text{there is } j \geq i : \sigma, j \models \psi \text{ and } \sigma, k \models \varphi \text{ for all } i \leq k < j$$

Linear temporal logic

LTL-formulas over a set of propositional variables AP generated by

$$\frac{}{p}\,p \in AP \qquad \frac{\varphi}{\neg\varphi} \qquad \frac{\varphi \quad \psi}{(\varphi \wedge \psi)} \qquad \frac{\varphi}{X\varphi} \qquad \frac{\varphi \quad \psi}{(\varphi U \psi)}$$

LTL semantics over $\sigma = a_0\ a_1\ \cdots \in \Sigma^\omega$ for $\Sigma := 2^{AP}$ and $i \in \mathbb{N}$:

$$\begin{array}{lcl} \sigma, i \models p & \iff & p \in a_i \\ \sigma, i \models \neg\varphi & \iff & \sigma, i \not\models \varphi \\ \sigma, i \models (\varphi \wedge \psi) & \iff & \sigma, i \models \varphi \text{ and } \sigma, i \models \psi \end{array}$$

Next

$$\sigma, i \models X\varphi \quad \iff \quad \sigma, i+1 \models \varphi$$

Until

$$\sigma, i \models (\varphi U \psi) \quad \iff \quad \text{there is } j \geq i : \sigma, j \models \psi \text{ and } \sigma, k \models \varphi \text{ for all } i \leq k < j$$

φ defines $L_\omega(\varphi) := \{\sigma \in \Sigma^\omega \mid \sigma, 0 \models \varphi\}$

Linear temporal logic

- Connectives $\vee, \rightarrow, \ldots$ defined as usual; $\bot := (p \wedge \neg p)$; $\top := \neg\bot$.
- Eventually $\Diamond\varphi := \top U \varphi$

  $\sigma, i \models \Diamond\varphi \iff$ there is $j \geq i : \sigma, j \models \varphi$
- Always $\Box\varphi := \neg\Diamond\neg\varphi$

  $\sigma, i \models \Box\varphi \iff$ for all $j \geq i : \sigma, j \models \varphi$

Linear temporal logic

- Connectives $\vee, \rightarrow, \ldots$ defined as usual; $\bot := (p \wedge \neg p)$; $\top := \neg\bot$.
- Eventually $\quad \Diamond\varphi := \top U \varphi$

  $\sigma, i \models \Diamond\varphi \iff$ there is $j \geq i : \sigma, j \models \varphi$
- Always $\quad \Box\varphi := \neg\Diamond\neg\varphi$

  $\sigma, i \models \Box\varphi \iff$ for all $j \geq i : \sigma, j \models \varphi$

Exercise $\varphi \equiv \psi$ means $L_\omega(\varphi) = L_\omega(\psi)$.

Dualities: $\neg\Box p \equiv \Diamond\neg p, \quad \neg X p \equiv X \neg p$

Idempotencies: $\Box\Box p \equiv \Box p, \quad pU(pUq) \equiv pUq, \qquad (pUq)Uq \equiv pUq$

Absorption laws: $\Diamond\Box\Diamond p \equiv \Box\Diamond p, \quad \Box\Diamond\Box p \equiv \Diamond\Box p$

Distributive laws: $\Box(p \wedge q) \equiv \Box p \wedge \Box q, \quad \Diamond(p \vee q) \equiv \Diamond p \vee \Diamond q, \quad X(pUq) \equiv Xp\ U\ Xq$

Expansion law: $pUq \equiv q \vee (p \wedge X(pUq))$

Linear temporal logic

- Connectives $\vee, \rightarrow, \ldots$ defined as usual; $\bot := (p \wedge \neg p)$; $\top := \neg\bot$.
- Eventually $\quad \Diamond\varphi := \top U \varphi$

  $$\sigma, i \models \Diamond\varphi \iff \text{there is } j \geq i : \sigma, j \models \varphi$$

- Always $\quad \Box\varphi := \neg\Diamond\neg\varphi$

  $$\sigma, i \models \Box\varphi \iff \text{for all } j \geq i : \sigma, j \models \varphi$$

Example imagine a transition system including a traffic light: propositional variables $g, y, r \in AP$ indicating "green", "yellow", "red".

"Once red, the light turns eventually green"

$$\Box(r \rightarrow \Diamond g)$$

"Once red, the light turns eventually green after being yellow for some time"

$$\Box\Big(r \rightarrow rU\big(y \wedge X(yUg)\big)\Big)$$

LTL versus FO

Theorem (Kamp 1968)

There are computable functions

$$\varphi \mapsto \varphi^{\mathsf{fo}} \quad \text{and} \quad \psi \mapsto \psi^{\mathsf{ltl}}$$

from LTL-formulas to FO-formulas and back, such that

$$L_\omega(\varphi) = L_\omega(\varphi^{\mathsf{fo}}) \quad \text{and} \quad L_\omega(\psi) = L_\omega(\psi^{\mathsf{ltl}})$$

LTL versus FO

Theorem (Kamp 1968)

There are computable functions

$$\varphi \mapsto \varphi^{\mathsf{fo}} \quad \text{and} \quad \psi \mapsto \psi^{\mathsf{ltl}}$$

from LTL-formulas to FO-formulas and back, such that

$$L_\omega(\varphi) = L_\omega(\varphi^{\mathsf{fo}}) \quad \text{and} \quad L_\omega(\psi) = L_\omega(\psi^{\mathsf{ltl}})$$

Proof of the easy part: let AP be the Boolean variables in φ and $\Sigma := 2^{AP}$.
Define $\varphi \mapsto \varphi^*(x)$ from LTL-to FO-formulas such that for all $\sigma \in \Sigma^\omega$ and $i \in \mathbb{N}$

$$\sigma, i \models_{\mathsf{LTL}} \varphi \quad \Longleftrightarrow \quad \mathcal{S}(\sigma) \models_{\mathsf{FO}} \varphi^*(i).$$

$$\begin{aligned}
&p^* := \textstyle\bigvee_{p \in a \in \Sigma} P_a(x),\\
&(\neg\varphi)^* := \neg\varphi^*(x),\\
&(\varphi \wedge \psi)^* := \varphi^*(x) \wedge \psi^*(x)\\
&(X\varphi)^* := \exists y\big(x \leq y \wedge \neg x = y \wedge \forall z(z \leq x \vee y \leq z) \wedge \varphi^*(y)\big),\\
&(\varphi U \psi)^* := \exists y\big(x \leq y \wedge \psi^*(y) \wedge \forall z(x \leq z \wedge z \leq y \wedge \neg z = y \rightarrow \varphi^*(z))\big). \quad \square
\end{aligned}$$

LTL Model-Checking

Theorem (Vardi, Wolper 1994)

There is a computable function that maps every LTL-formula φ to an NBA $\mathbb{A}_\varphi$ of size $2^{O(|\varphi|)}$ such that

$$L_\omega(\varphi) = L_\omega(\mathbb{A}_\varphi).$$

LTL Model-Checking

Theorem (Vardi, Wolper 1994)

There is a computable function that maps every LTL-formula φ to an NBA $\mathbb{A}_\varphi$ of size $2^{O(|\varphi|)}$ such that

$$L_\omega(\varphi) = L_\omega(\mathbb{A}_\varphi).$$

Proof

Γ := set of subformulas of φ,
AP := set of Boolean variables in φ,
$\Sigma := 2^{AP}$.

$\Gamma_i^\sigma := \{\psi \in \Gamma \mid \sigma, i \models \psi\}$ where $\sigma \in \Sigma^\omega$ and $i \in \mathbb{N}$

This is a type: a set $s \subseteq \Gamma$ such that for formulas in Γ

$$\psi_0 \wedge \psi_1 \in s \Longleftrightarrow \psi_0 \in s \text{ and } \psi_1 \in s,$$

$$\neg\psi \in s \Longleftrightarrow \psi \notin s,$$

$$\psi_1 \in s \Longrightarrow \psi_0 U \psi_1 \in s \Longrightarrow \psi_0 \in s \text{ or } \psi_1 \in s.$$

Let S denote the set of types.

LTL Model-Checking

Claim Types can be computed by an automaton: there are $\Delta \subseteq S \times \Sigma \times S$, $\mathcal{F} \subseteq 2^S$ of size $|\varphi|$ such that for all $s_0 \in S$, $\sigma = a_0 a_1 \cdots \Sigma^\omega$ tfae:

(a) $s_0\ a_0\ s_1\ a_1 \cdots$ is an accepting run of the GNBA $(S, s_0, \Sigma, \Delta, \mathcal{F})$

(b) $\Gamma_i^\sigma = s_i$ for all $i \in \mathbb{N}$.

LTL Model-Checking

Claim Types can be computed by an automaton: there are $\Delta \subseteq S \times \Sigma \times S$, $\mathcal{F} \subseteq 2^S$ of size $|\varphi|$ such that for all $s_0 \in S$, $\sigma = a_0 a_1 \cdots \Sigma^\omega$ tfae:

(a) $s_0 \ a_0 \ s_1 \ a_1 \cdots$ is an accepting run of the GNBA $(S, s_0, \Sigma, \Delta, \mathcal{F})$

(b) $\Gamma_i^\sigma = s_i$ for all $i \in \mathbb{N}$.

Suffices!

Consider GNBA $\mathbb{A} := (S \cup \{s_0^*\}, \Sigma, s_0^*, \Delta^*, \mathcal{F})$ with new s_0^* and

$\Delta^* := \Delta \cup \big\{(s_0^*, a, s) \mid \text{exists } s_0 \in S : \varphi \in s_0 \text{ and } (s_0, a, s) \in \Delta\big\}$

satisfies $L_\omega(\mathbb{A}) = L_\omega(\varphi)$.

Exercise gives equivalent NBA $\mathbb{A}_\varphi$ with $|\mathcal{F}| \cdot |S \cup \{s_0^*\}| \leq 2^{2|\varphi|}$ many states.

LTL Model-Checking

Claim Types can be computed by an automaton: there are $\Delta \subseteq S \times \Sigma \times S$, $\mathcal{F} \subseteq 2^S$ of size $|\varphi|$ such that for all $s_0 \in S$, $\sigma = a_0 a_1 \cdots \Sigma^\omega$ tfae:

(a) $s_0\ a_0\ s_1\ a_1 \cdots$ is an accepting run of the GNBA $(S, s_0, \Sigma, \Delta, \mathcal{F})$

(b) $\Gamma_i^\sigma = s_i$ for all $i \in \mathbb{N}$.

Δ contains (s, a, s') iff $a = s \cap AP$ and for formulas in Γ

$X\psi \in s \Longleftrightarrow \psi \in s'$

$\psi_0 U \psi_1 \in s \Longleftrightarrow \psi_1 \in s$ or, both $\psi_0 \in s$ and $\psi_0 U \psi_1 \in s'$

$\mathcal{F}$ contains for every $\psi_0 U \psi_1 \in \Gamma$ the set $\{s \mid \psi_0 U \psi_1 \notin s\} \cup \{s \mid \psi_1 \in s\}$

LTL Model-Checking

Claim Types can be computed by an automaton: there are $\Delta \subseteq S \times \Sigma \times S$, $\mathcal{F} \subseteq 2^S$ of size $|\varphi|$ such that for all $s_0 \in S$, $\sigma = a_0 a_1 \cdots \Sigma^\omega$ tfae:

(a) $s_0\ a_0\ s_1\ a_1 \cdots$ is an accepting run of the GNBA $(S, s_0, \Sigma, \Delta, \mathcal{F})$

(b) $\Gamma_i^\sigma = s_i$ for all $i \in \mathbb{N}$.

Δ contains (s, a, s') iff $a = s \cap AP$ and for formulas in Γ

$X\psi \in s \Longleftrightarrow \psi \in s'$

$\psi_0 U \psi_1 \in s \Longleftrightarrow \psi_1 \in s$ or, both $\psi_0 \in s$ and $\psi_0 U \psi_1 \in s'$

$\mathcal{F}$ contains for every $\psi_0 U \psi_1 \in \Gamma$ the set $\{s \mid \psi_0 U \psi_1 \notin s\} \cup \{s \mid \psi_1 \in s\}$

Fix $s_0 \in S$. (b) $\Rightarrow$ (a) easy. For (a) $\Rightarrow$ (b) show

$$\sigma, i \models \psi \Longleftrightarrow \psi \in s_i$$

for all $\psi \in \Gamma$ by induction on ψ. Case $\psi = \psi_0 U \psi_1$. For simplicity $i = 0$. Show:

$$\sigma, 0 \models \psi_0 U \psi_1 \iff \psi_0 U \psi_1 \in s_0.$$

LTL Model-Checking

$\triangle$ $\psi_0 U \psi_1 \in s \Longleftrightarrow \psi_1 \in s$ or, both $\psi_0 \in s$ and $\psi_0 U \psi_1 \in s'$

$\mathcal{F} \ni \{s \mid \psi_0 U \psi_1 \notin s\} \cup \{s \mid \psi_1 \in s\}$

Want $\sigma, 0 \models \psi_0 U \psi_1 \iff \psi_0 U \psi_1 \in s_0.$

LTL Model-Checking

Δ $\quad \psi_0 U \psi_1 \in s \Longleftrightarrow \psi_1 \in s$ or, both $\psi_0 \in s$ and $\psi_0 U \psi_1 \in s'$

$\mathcal{F} \ni \{s \mid \psi_0 U \psi_1 \notin s\} \cup \{s \mid \psi_1 \in s\}$

Want $\quad \sigma, 0 \models \psi_0 U \psi_1 \iff \psi_0 U \psi_1 \in s_0$.

$\Rightarrow$ choose i such that: $\sigma, i \models \psi_1$ and $\sigma, j \models \psi_0$ for all $j < i$

by induction: $\psi_1 \in s_i$ and $\psi_0 \in s_j$ for all $j < i$

by type-definition: $\psi_0 U \psi_1 \in s_i$

as $\psi_0 \in s_{i-1}$, by Δ-definition: $\psi_0 U \psi_1 \in s_{i-1}$

continue... $\psi_0 U \psi_1 \in s_0$.

LTL Model-Checking

Δ $\quad \psi_0 U \psi_1 \in s \Longleftrightarrow \psi_1 \in s$ or, both $\psi_0 \in s$ and $\psi_0 U \psi_1 \in s'$

$\mathcal{F} \ni \{s \mid \psi_0 U \psi_1 \notin s\} \cup \{s \mid \psi_1 \in s\}$

Want $\quad \sigma, 0 \models \psi_0 U \psi_1 \iff \psi_0 U \psi_1 \in s_0$.

$\Leftarrow$ by type-definition: $\psi_0 \in s_0$ or $\psi_1 \in s_0$

if $\psi_1 \in s_0$: by induction $\sigma, 0 \models \psi_1$, so $\sigma, 0 \models \psi_0 U \psi_1$ done!

else $\psi_0 \in s_0 \not\ni \psi_1$: by Δ-definition $\psi_0 U \psi_1 \in s_1$

by type-definition: $\psi_0 \in s_1$ or $\psi_1 \in s_1$

if $\psi_1 \in s_1$: by induction $\sigma, 1 \models \psi_1$ and $\sigma, 0 \models \psi_0$, so $\sigma, 0 \models \psi_0 U \psi_1$ done!

else $\psi_0 \in s_1 \not\ni \psi_1$: by Δ-definition $\psi_0 U \psi_1 \in s_2$

... continue until $\psi_0 U \psi_1 \in s_j \in \{s \mid \psi_0 U \psi_1 \notin s\} \cup \{s \mid \psi_1 \in s\}$.

Then $\psi_1 \in s_j$: done! $\square$

LTL Model-Checking

As before:

Corollary The problem

> Input: a transition system $\mathbb{T}$, an LTL-formula φ.
>
> Problem: $\sigma, 0 \models \varphi$ for every trace σ of $\mathbb{T}$?

is decidable in time $2^{O(|\varphi|)} \cdot |\mathbb{T}|$.

LTL Model-Checking

Proposition

LTL-formulas φ do not have equivalent NBAs with $2^{o(\sqrt{|\varphi|})}$ states.

Proof Let $AP = \{p\}$. For $n \in \mathbb{N}$ let L_n contain the words

$$a_0 \ a_1 \cdots a_{n-1} \ a_0 \ a_1 \cdots a_{n-1} \ \emptyset \ \emptyset \cdots$$

Defined by the size $O(n^2)$ formula

$$\bigwedge_{i<n} (X^i p \leftrightarrow X^{n+i} p).$$

Let $\mathbb{A}$ be an NBA with $L_\omega(\mathbb{A}) = L_n$. For each $a_0 \cdots a_{n-1} \in 2^{AP}$ there is a state $q(a_0 \cdots a_{n-1})$ such that $\mathbb{A}$ with this starting state accepts

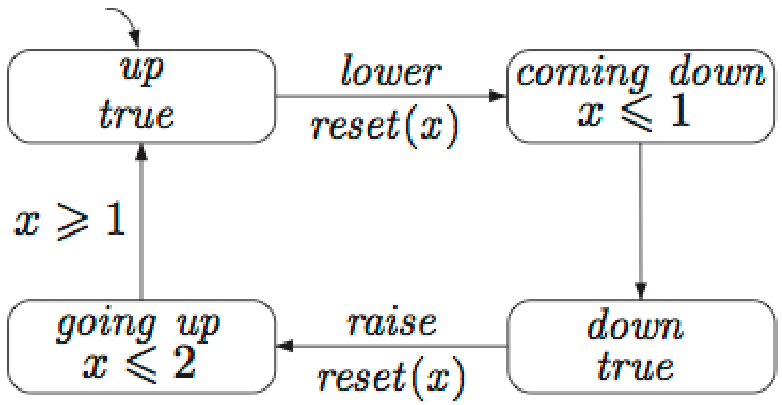
$$a_0 \ a_1 \cdots a_{n-1} \ \emptyset \ \emptyset \cdots .$$

This is not true for $q(a'_0 \cdots a'_{n-1})$ for every $a'_0 \cdots a'_{n-1} \neq a_0 \cdots a_{n-1}$.

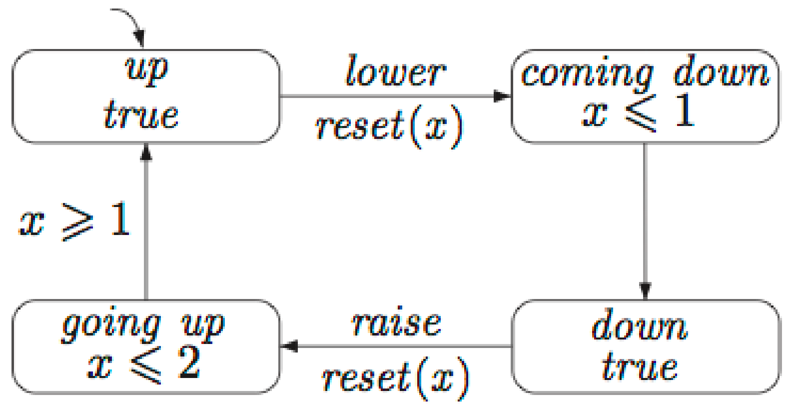
Hence the states $q(a_0 \cdots a_{n-1})$ are pairwise distinct.

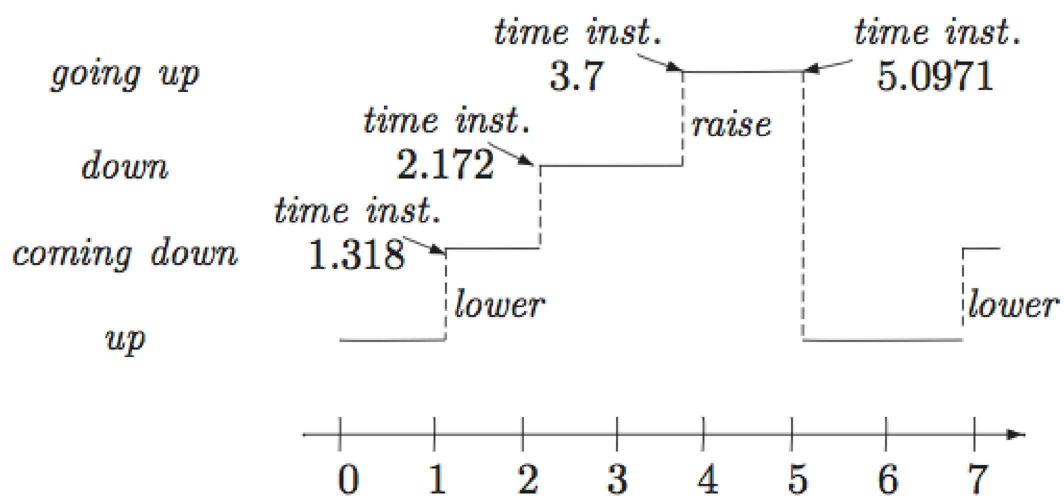
Hence $\mathbb{A}$ has at least 2^n many states. $\square$

Timed automata



Timed automata





Timed automata

Timed automaton $\mathbb{A}$ consists of:

- Loc finite set of locations
- $Loc_0 \subseteq Loc$ initial locations
- Act finite set of actions
- AP finite set of propositional variables
- $L : Loc \to 2^{AP}$ a labeling

Timed automata

Timed automaton $\mathbb{A}$ consists of:

Loc finite set of locations

$Loc_0 \subseteq Loc$ initial locations

Act finite set of actions

AP finite set of propositional variables

$L : Loc \to 2^{AP}$ a labeling

C finite set of clocks

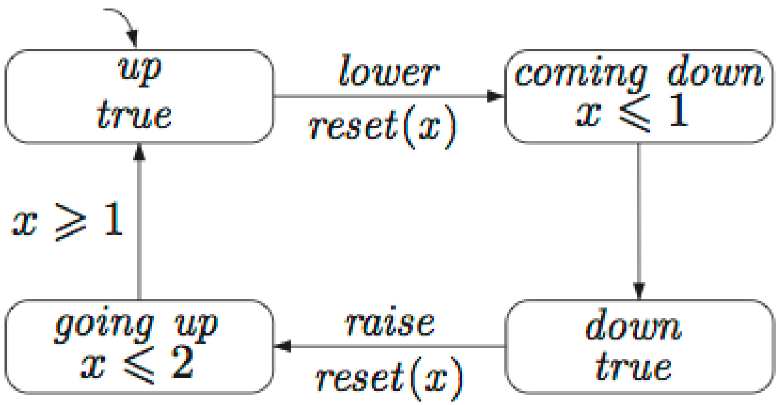
$Inv : Loc \to CC$:= finite sets of clock constraints

$$x \sim k \text{ where } \begin{array}{l} x \in C \\ k \in \mathbb{N} \\ \sim \in \{<, \leq, =, >, \geq\} \end{array}$$

$\hookrightarrow \subseteq Loc \times CC \times Act \times 2^C \times Loc$

View $(\ell, g, \alpha, X, \ell') \in \hookrightarrow$ as an arrow from ℓ to ℓ' labeled by a guard $g \in CC$, an action $\alpha \in Act$ and a set of clocks $X \subseteq C$ that are reset.

Timed automata: example

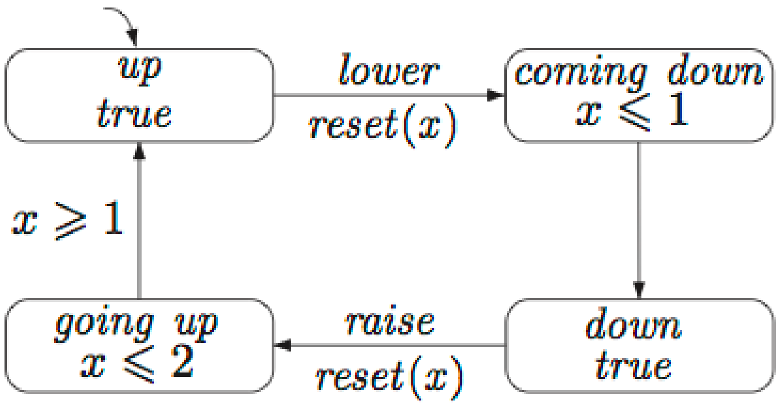


$Loc = AP = \{up, down, goingup, comingdown\}, \quad Loc_0 = \{up\}, \quad L(\ell) = \{\ell\}$

$C = \{x\}$

$Act = \{raise, lower, \tau\}$

Timed automata: example



$Loc = AP = \{up, down, goingup, comingdown\}, \quad Loc_0 = \{up\}, \quad L(\ell) = \{\ell\}$

$C = \{x\}$

$Act = \{raise, lower, \tau\}$

$Inv(up) = \emptyset$

$Inv(comingdown) = \{x \leq 1\}$

$(up, \emptyset, lower, \{x\}, comingdown) \in \hookrightarrow$

$(goingup, \{x \geq 1\}, \tau, \emptyset, up) \in \hookrightarrow$

Transition system $TS(\mathbb{A})$ of $\mathbb{A}$:

states: $\langle \ell, \eta \rangle$ with $\ell \in Loc$ and $\eta : C \to \mathbb{R}_{\geq 0}$ a clock valuation

initial states: $\langle \ell, \eta \rangle$ with $\ell \in Loc_0$ and η is constantly 0.

propositional variables: $AP \ \dot{\cup}$ clock constraints

labeling of $\langle \ell, \eta \rangle$ is $L(\ell) \cup \{ x{\sim}k \mid \eta(x) \sim k, x \in C \}$

actions: $Act \ \dot{\cup} \ \mathbb{R}_{\geq 0}$

Transition system $TS(\mathbb{A})$ of $\mathbb{A}$:

states: $\langle \ell, \eta \rangle$ with $\ell \in Loc$ and $\eta : C \to \mathbb{R}_{\geq 0}$ a clock valuation

initial states: $\langle \ell, \eta \rangle$ with $\ell \in Loc_0$ and η is constantly 0.

propositional variables: $AP \mathbin{\dot{\cup}}$ clock constraints

labeling of $\langle \ell, \eta \rangle$ is $L(\ell) \cup \left\{ x{\sim}k \mid \eta(x) \sim k, x \in C \right\}$

actions: $Act \mathbin{\dot{\cup}} \mathbb{R}_{\geq 0}$

discrete transitions $\langle \ell, \eta \rangle \xrightarrow{\alpha} \langle \ell', \eta' \rangle$ where $\alpha \in Act$ and

- $(\ell, g, \alpha, X, \ell') \in \hookrightarrow$
- η satisfies g (ie. $\eta(x) \sim k$ for all $x{\sim}k \in g$)
- $\eta'(x) = \begin{cases} \eta(x) & x \notin X \\ 0 & x \in X \end{cases}$
- η' satisfies $Inv(\ell)$

delay transitions $\langle \ell, \eta \rangle \xrightarrow{d} \langle \ell, \eta + d \rangle$ where $d \in \mathbb{R}_{\geq 0}$ and

- $(\eta + d)(x) = \eta(x) + d$ for all $x \in C$
- $\eta + d'$ satisfies $Inv(\ell)$ for all $d' \in [0, d]$.

Executing timed automata

- Relevant paths starting at $\langle \ell_0, \eta_0 \rangle$ have the form

$$\langle \ell_0, \eta_0 \rangle \xrightarrow{d_0} \langle \ell_0, \eta_0 + d_0 \rangle \xrightarrow{\alpha_0} \langle \ell_1, \eta_1 \rangle \xrightarrow{d_1} \langle \ell_1, \eta_1 + d_1 \rangle \xrightarrow{\alpha_1} \langle \ell_2, \eta_2 \rangle \xrightarrow{d_2} \cdots$$

where $\alpha_i \in Act, d_i \in \mathbb{R}_{\geq 0}$ and $\sum_i d_i = \infty$, or:

$$\langle \ell_0, \eta_0 \rangle \xrightarrow{d_0} \cdots \xrightarrow{\alpha_{k-1}} \langle \ell_k, \eta_k \rangle \xrightarrow{d_k} \langle \ell_k, \eta_k + 1 \rangle \xrightarrow{d_{k+1}} \langle \ell_k, \eta_k + 2 \rangle \xrightarrow{d_{k+2}} \cdots$$

where $1 = d_k = d_{k+1} = \ldots$

Executing timed automata

- Relevant paths starting at $\langle \ell_0, \eta_0 \rangle$ have the form

$$\langle \ell_0, \eta_0 \rangle \xrightarrow{d_0} \langle \ell_0, \eta_0 + d_0 \rangle \xrightarrow{\alpha_0} \langle \ell_1, \eta_1 \rangle \xrightarrow{d_1} \langle \ell_1, \eta_1 + d_1 \rangle \xrightarrow{\alpha_1} \langle \ell_2, \eta_2 \rangle \xrightarrow{d_2} \cdots$$

where $\alpha_i \in Act, d_i \in \mathbb{R}_{\geq 0}$ and $\sum_i d_i = \infty$, or:

$$\langle \ell_0, \eta_0 \rangle \xrightarrow{d_0} \cdots \xrightarrow{\alpha_{k-1}} \langle \ell_k, \eta_k \rangle \xrightarrow{d_k} \langle \ell_k, \eta_k + 1 \rangle \xrightarrow{d_{k+1}} \langle \ell_k, \eta_k + 2 \rangle \xrightarrow{d_{k+2}} \cdots$$

where $1 = d_k = d_{k+1} = \ldots$

- $\langle \ell, \eta \rangle$ is at time $t \in \mathbb{R}_{\geq 0}$ in a relevant path π iff

  there are $i \in \mathbb{N}, d \in [0, d_i]$ such that $\ell = \ell_i$ and $\eta = \eta_i + d$ and $t = \sum_{j \leq i} d_j + d$.

Executing timed automata

- Relevant paths starting at $\langle \ell_0, \eta_0 \rangle$ have the form

$$\langle \ell_0, \eta_0 \rangle \xrightarrow{d_0} \langle \ell_0, \eta_0 + d_0 \rangle \xrightarrow{\alpha_0} \langle \ell_1, \eta_1 \rangle \xrightarrow{d_1} \langle \ell_1, \eta_1 + d_1 \rangle \xrightarrow{\alpha_1} \langle \ell_2, \eta_2 \rangle \xrightarrow{d_2} \cdots$$

where $\alpha_i \in Act, d_i \in \mathbb{R}_{\geq 0}$ and $\sum_i d_i = \infty$, or:

$$\langle \ell_0, \eta_0 \rangle \xrightarrow{d_0} \cdots \xrightarrow{\alpha_{k-1}} \langle \ell_k, \eta_k \rangle \xrightarrow{d_k} \langle \ell_k, \eta_k + 1 \rangle \xrightarrow{d_{k+1}} \langle \ell_k, \eta_k + 2 \rangle \xrightarrow{d_{k+2}} \cdots$$

where $1 = d_k = d_{k+1} = \ldots$

- $\langle \ell, \eta \rangle$ is at time $t \in \mathbb{R}_{\geq 0}$ in a relevant path π iff

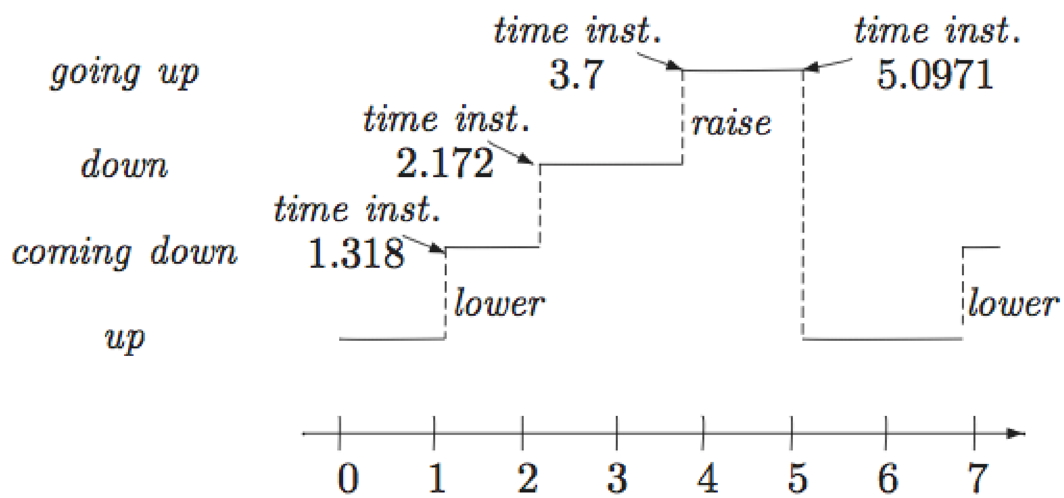
  there are $i \in \mathbb{N}, d \in [0, d_i]$ such that $\ell = \ell_i$ and $\eta = \eta_i + d$ and $t = \sum_{j \leq i} d_j + d$.

- Then $\langle \ell', \eta' \rangle$ before $\langle \ell, \eta \rangle$ iff $\langle \ell, \eta \rangle$ after $\langle \ell', \eta' \rangle$ iff

  $\langle \ell', \eta' \rangle$ is at some time $t' < t$ in π, or,
  $\langle \ell', \eta' \rangle$ is at time t in π and $\ell' = \ell_j$ for some $j < i$.

E.g. above: $\langle \ell_1, \eta_1 \rangle$ is at time d_0, and $\langle \ell_0, \eta_0 + d_0 \rangle$ too and before $\langle \ell_1, \eta_1 \rangle$.

Executing timed automata: example



write $\eta(x) \in \mathbb{R}_{\geq 0}$ instead $\eta : \{x\} \to \mathbb{R}_{\geq 0}$:

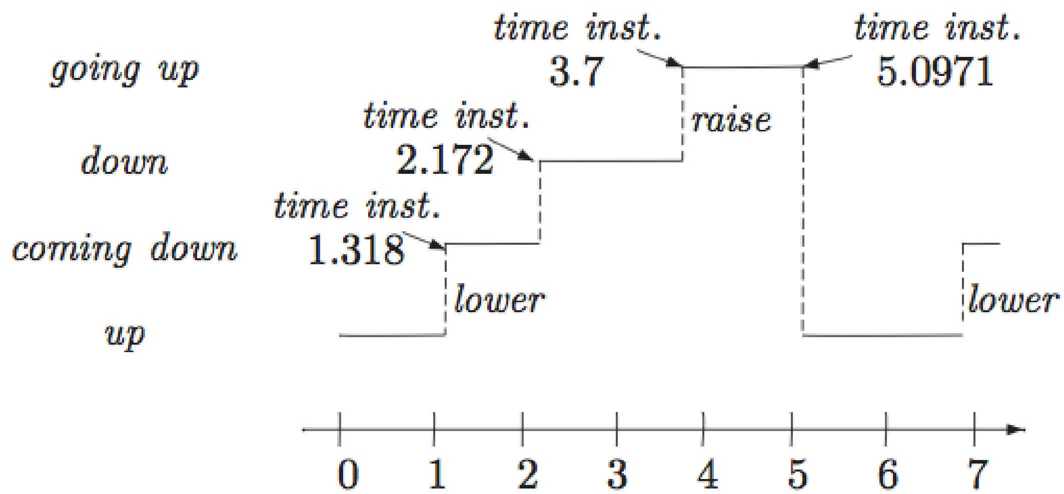
$\langle up, 1.318 \rangle$ at time 1.318

before $\langle comingdown, 0 \rangle$ at time 1.318

before $\langle comingdown, 0.002 \rangle$ at time 1.32

before $\langle goingup, 0.5 \rangle$ is at time 4.2

Executing timed automata: example



$$\langle up, 0\rangle \xrightarrow{1.318} \langle up, 1.318\rangle \xrightarrow{lower} \langle comingdown, 0\rangle \xrightarrow{0.854} \langle comingdown, 0.854\rangle$$

$$\xrightarrow{\tau} \langle down, 0.854\rangle \xrightarrow{1.528} \langle down, 2.382\rangle \xrightarrow{raise} \langle goingup, 0\rangle \xrightarrow{1.3971} \langle goingup, 1.3971\rangle$$

$$\ldots$$

Timed computation tree logic

TCTL-formulas over $AP \cup CC$ generated by

$$\frac{}{p} \qquad \frac{\varphi}{\neg\varphi} \qquad \frac{\varphi \quad \psi}{(\varphi \wedge \psi)} \qquad \frac{\varphi \quad \psi}{\forall(\varphi\ U_I\ \psi)} \qquad \frac{\varphi \quad \psi}{\exists(\varphi\ U_I\ \psi)}$$

where $p \in AP \cup CC$, I an interval $[a, b), (a, b], (a, b), [a, b]$ for $a \leq b$ in $\mathbb{N} \cup \{\infty\}$.

Timed computation tree logic

TCTL-formulas over $AP \cup CC$ generated by

$$\frac{}{p} \qquad \frac{\varphi}{\neg\varphi} \qquad \frac{\varphi \quad \psi}{(\varphi \wedge \psi)} \qquad \frac{\varphi \quad \psi}{\forall(\varphi\ U_I\ \psi)} \qquad \frac{\varphi \quad \psi}{\exists(\varphi\ U_I\ \psi)}$$

where $p \in AP \cup CC$, I an interval $[a,b), (a,b], (a,b), [a,b]$ for $a \leq b$ in $\mathbb{N} \cup \{\infty\}$.

TCTL semantics over $TS(\mathbb{A})$:

$$\begin{array}{lcll}
\langle \ell, \eta \rangle \models p & \iff & p \in L(\ell) & \text{for } p \in AP \\
\langle \ell, \eta \rangle \models x{\sim}k & \iff & \eta(x) \sim k & \text{for } x{\sim}k \in CC \\
\langle \ell, \eta \rangle \models \neg\varphi & \iff & \langle \ell, \eta \rangle \not\models \varphi & \\
\langle \ell, \eta \rangle \models \varphi \wedge \psi & \iff & \langle \ell, \eta \rangle \models \varphi \text{ and } \langle \ell, \eta \rangle \models \psi &
\end{array}$$

Timed computation tree logic

TCTL-formulas over $AP \cup CC$ generated by

$$\frac{}{p} \qquad \frac{\varphi}{\neg\varphi} \qquad \frac{\varphi \quad \psi}{(\varphi \wedge \psi)} \qquad \frac{\varphi \quad \psi}{\forall(\varphi\ U_I\ \psi)} \qquad \frac{\varphi \quad \psi}{\exists(\varphi\ U_I\ \psi)}$$

where $p \in AP \cup CC$, I an interval $[a,b), (a,b], (a,b), [a,b]$ for $a \leq b$ in $\mathbb{N} \cup \{\infty\}$.

TCTL semantics over $TS(\mathbb{A})$:

$$\begin{array}{lcll}
\langle \ell, \eta \rangle \models p & \iff & p \in L(\ell) & \text{for } p \in AP \\
\langle \ell, \eta \rangle \models x{\sim}k & \iff & \eta(x) \sim k & \text{for } x{\sim}k \in CC \\
\langle \ell, \eta \rangle \models \neg\varphi & \iff & \langle \ell, \eta \rangle \not\models \varphi & \\
\langle \ell, \eta \rangle \models \varphi \wedge \psi & \iff & \langle \ell, \eta \rangle \models \varphi \text{ and } \langle \ell, \eta \rangle \models \psi &
\end{array}$$

$\langle \ell, \eta \rangle \models \forall(\varphi\ U_I\ \psi) \quad \iff$

for every relevant path π starting at $\langle \ell, \eta \rangle$

there is a state $\langle \ell', \eta' \rangle$ at some time $t \in I$ in π such that

$\langle \ell', \eta' \rangle \models \psi$ and

$\langle \ell'', \eta'' \rangle \models \varphi \vee \psi$ for all states $\langle \ell'', \eta'' \rangle$ before $\langle \ell', \eta' \rangle$.

Timed computation tree logic

TCTL-formulas over $AP \cup CC$ generated by

$$\frac{}{p} \qquad \frac{\varphi}{\neg\varphi} \qquad \frac{\varphi \quad \psi}{(\varphi \wedge \psi)} \qquad \frac{\varphi \quad \psi}{\forall(\varphi\ U_I\ \psi)} \qquad \frac{\varphi \quad \psi}{\exists(\varphi\ U_I\ \psi)}$$

where $p \in AP \cup CC$, I an interval $[a,b),(a,b],(a,b),[a,b]$ for $a \leq b$ in $\mathbb{N} \cup \{\infty\}$.

TCTL semantics over $TS(\mathbb{A})$:

$$\begin{array}{lcll}
\langle \ell, \eta \rangle \models p & \iff & p \in L(\ell) & \text{for } p \in AP \\
\langle \ell, \eta \rangle \models x{\sim}k & \iff & \eta(x) \sim k & \text{for } x{\sim}k \in CC \\
\langle \ell, \eta \rangle \models \neg\varphi & \iff & \langle \ell, \eta \rangle \not\models \varphi & \\
\langle \ell, \eta \rangle \models \varphi \wedge \psi & \iff & \langle \ell, \eta \rangle \models \varphi \text{ and } \langle \ell, \eta \rangle \models \psi &
\end{array}$$

$\langle \ell, \eta \rangle \models \exists(\varphi\ U_I\ \psi) \iff$

for some relevant path π starting at $\langle \ell, \eta \rangle$

there is a state $\langle \ell', \eta' \rangle$ at some time $t \in I$ in π such that

$\langle \ell', \eta' \rangle \models \psi$ and
$\langle \ell'', \eta'' \rangle \models \varphi \vee \psi$ for all states $\langle \ell'', \eta'' \rangle$ before $\langle \ell', \eta' \rangle$.

Timed computation tree logic

$\forall\Diamond_I \varphi := \forall(\top \ U_I \ \varphi)$

$\langle \ell, \eta \rangle \models \forall\Diamond_I \varphi$ iff for every relevant path π starting at $\langle \ell, \eta \rangle$
there is a state $\langle \ell', \eta' \rangle$ at some time $t \in I$ in π such that $\langle \ell', \eta' \rangle \models \varphi$

$\exists\Diamond_I \varphi := \exists(\top \ U_I \ \varphi)$

$\langle \ell, \eta \rangle \models \forall\Diamond_I \varphi$ iff for some relevant path π starting at $\langle \ell, \eta \rangle$
there is a state $\langle \ell', \eta' \rangle$ at some time $t \in I$ in π such that $\langle \ell', \eta' \rangle \models \varphi$

Timed computation tree logic

$\forall\Diamond_I\varphi := \forall(\top\ U_I\ \varphi)$

$\langle \ell, \eta\rangle \models \forall\Diamond_I\varphi$ iff for every relevant path π starting at $\langle \ell, \eta\rangle$
there is a state $\langle \ell', \eta'\rangle$ at some time $t \in I$ in π such that $\langle \ell', \eta'\rangle \models \varphi$

$\exists\Diamond_I\varphi := \exists(\top\ U_I\ \varphi)$

$\langle \ell, \eta\rangle \models \forall\Diamond_I\varphi$ iff for some relevant path π starting at $\langle \ell, \eta\rangle$
there is a state $\langle \ell', \eta'\rangle$ at some time $t \in I$ in π such that $\langle \ell', \eta'\rangle \models \varphi$

$\forall\Box_I\varphi := \neg\exists\Diamond_I\neg\varphi$

$\langle \ell, \eta\rangle \models \forall\Box_I\varphi$ iff for every relevant path π starting at $\langle \ell, \eta\rangle$
for all states $\langle \ell', \eta'\rangle$ at some time $t \in I$ in π: $\langle \ell', \eta'\rangle \models \varphi$

$\exists\Box_I\varphi := \neg\forall\Diamond_I\neg\varphi$

$\langle \ell, \eta\rangle \models \exists\Box_I\varphi$ iff for some relevant path π starting at $\langle \ell, \eta\rangle$
for all states $\langle \ell', \eta'\rangle$ at some time $t \in I$ in π: $\langle \ell', \eta'\rangle \models \varphi$

Omit subscript $I = [0, \infty)$.

Timed computation tree logic

Leads-to operator

$\varphi \leadsto \psi := \forall\Box(\varphi \rightarrow \forall\Diamond\psi)$

$\langle \ell, \eta \rangle \models \varphi \leadsto \psi$

iff

for every relevant path π starting at $\langle \ell, \eta \rangle$

for every state $\langle \ell', \eta' \rangle \models \varphi$ at some time $t \in [0, \infty)$ in π:

for all relevant paths π' starting at $\langle \ell', \eta' \rangle$

there are a state $\langle \ell'', \eta'' \rangle$ at some time $t' \in [0, \infty)$ in π' st $\langle \ell'', \eta'' \rangle \models \psi$

iff

for every relevant path π starting at $\langle \ell, \eta \rangle$ and every $t \in [0, \infty)$:

for every state $\langle \ell', \eta' \rangle \models \varphi$ at time t in π

there is a state $\langle \ell'', \eta'' \rangle \models \psi$ after $\langle \ell', \eta' \rangle$.

Timed computation tree logic

Elimination of time constraints

Assume there is a clock $z \in C$ that is never reset and assume $\eta(z) = 0$.

$$\langle \ell, \eta \rangle \models \forall (p \ U_{[a,b)} \ q) \quad \iff \quad \langle \ell, \eta \rangle \models \forall \big((p \vee q) \ U \ (z {\geq} \ a \wedge z {<} b \wedge q) \big)$$

$$\langle \ell, \eta \rangle \models \forall \Diamond_{[a,b)} q \quad \iff \quad \langle \ell, \eta \rangle \models \forall \Diamond (z {\geq} \ a \wedge z {<} b \wedge q)$$

$$\langle \ell, \eta \rangle \models \forall \Box_{[a,b)} q \quad \iff \quad \langle \ell, \eta \rangle \models \forall \Box (z {\geq} \ a \wedge z {<} b \rightarrow q)$$

Caution

trick useless for iterated modalities

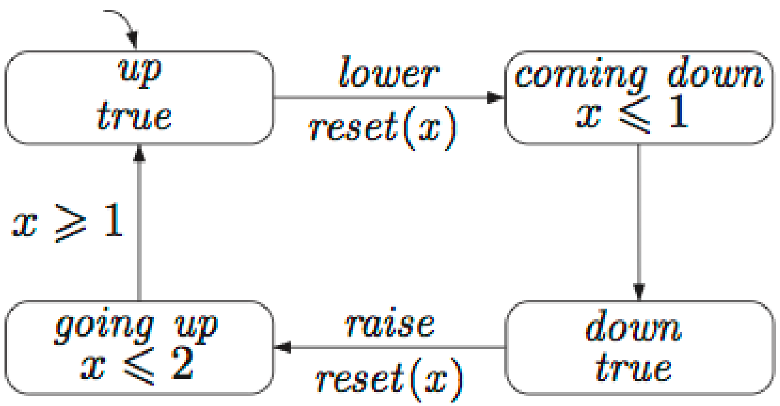
UPPAAL

supports

$$\forall \Box \varphi, \quad \forall \Diamond \varphi, \quad \exists \Box \varphi, \quad \forall \Diamond \varphi, \quad \varphi \leadsto \psi$$

for φ, ψ in propositional logic.

Timed computation tree logic: examples



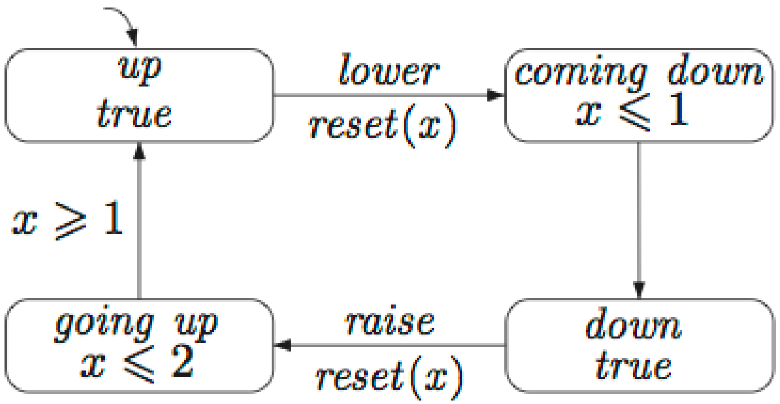
The initial state $\langle up, 0 \rangle$ of the associated transition system satisfies:

$\forall\Box(comingdown \rightarrow x{\leq}1)$

$comingdown \rightsquigarrow (down \wedge x{\leq}1)$

$goingup \rightsquigarrow (1{\leq}x \wedge x{\leq}2 \wedge up)$

Timed computation tree logic: examples



The initial state $\langle up, 0\rangle$ of the associated transition system satisfies:

$\forall\Box(comingdown \to x\leq 1)$

$comingdown \rightsquigarrow (down \wedge x\leq 1)$

$goingup \rightsquigarrow (1\leq x \wedge x\leq 2 \wedge up)$

$\forall\Box(goingup \wedge x=0 \to \forall\Diamond_{[1,2]} up)$

$\forall\Box(goingup \wedge x>1 \to \forall\Diamond_{[0,1)} up)$

Model-checking TCTL: theorem

$\mathbb{A}$ is timelock free iff at reachable states of $TS(\mathbb{A})$ start relevant paths.

Model-checking TCTL: theorem

$\mathbb{A}$ is timelock free iff at reachable states of $TS(\mathbb{A})$ start relevant paths.

Theorem (Alur, Courcoubetis, Dill 1990)

The promise problem

> Input: a timelock-free timed automaton $\mathbb{A}$, a TCTL-formula φ
> Problem: does every initial state of $TS(\mathbb{A})$ satisfy φ?

is decidable in time

$$k^{O(c)} \cdot |\varphi| \cdot |\mathbb{A}|$$

where c is the number of clocks in $\mathbb{A}$ and $k \geq c$ upper bounds the natural numbers appearing in φ and $\mathbb{A}$.

Model-checking TCTL: proof

Assume $\mathbb{A}$ has a clock $x_{\text{real}} \in C$ not mentioned in guards, invariants or φ.

For $r \in \mathbb{R}_{\geq 0}$ write $\langle r \rangle := r - \lfloor r \rfloor$. E.g., $\langle 1.23 \rangle = 0.23$.

$\eta \approx \eta'$ iff

- for all $x \in C$: $\lfloor \eta(x) \rfloor$ and $\lfloor \eta'(x) \rfloor$ are equal or both $\geq k$.
- for all $x, y \in C$ with $\eta(x) < k, \eta(y) < k$: $\begin{array}{l} \langle \eta(x) \rangle \leq \langle \eta(y) \rangle \text{ iff } \langle \eta'(x) \rangle \leq \langle \eta'(y) \rangle \\ \langle \eta(x) \rangle = 0 \text{ iff } \langle \eta'(x) \rangle = 0 \end{array}$

Claim: if $\eta \approx \eta'$ and $d \in \mathbb{R}_{\geq 0}$, then $\eta + d \approx \eta' + d'$ for some $d' \in \mathbb{R}_{\geq 0}$

Model-checking TCTL: proof

Assume $\mathbb{A}$ has a clock $x_{\text{real}} \in C$ not mentioned in guards, invariants or φ.

For $r \in \mathbb{R}_{\geq 0}$ write $\langle r \rangle := r - \lfloor r \rfloor$. E.g., $\langle 1.23 \rangle = 0.23$.

$\eta \approx \eta'$ iff

- for all $x \in C$: $\lfloor \eta(x) \rfloor$ and $\lfloor \eta'(x) \rfloor$ are equal or both $\geq k$.
- for all $x, y \in C$ with $\eta(x) < k, \eta(y) < k$: $\langle \eta(x) \rangle \leq \langle \eta(y) \rangle$ iff $\langle \eta'(x) \rangle \leq \langle \eta'(y) \rangle$; $\langle \eta(x) \rangle = 0$ iff $\langle \eta'(x) \rangle = 0$

Claim: if $\eta \approx \eta'$ and $d \in \mathbb{R}_{\geq 0}$, then $\eta + d \approx \eta' + d'$ for some $d' \in \mathbb{R}_{\geq 0}$

Suffices for $d < 1$ (then choose d'' for $\langle d \rangle$ and set $d' := \lfloor d \rfloor + d''$)

If there is $x \in C$ such that $d = 1 - \langle \eta(x) \rangle$, set $d' := 1 - \langle \eta'(x) \rangle$

Otw order clocks $x_1 \leq \cdots \leq x_c$ according $\langle \eta(x) \rangle$ (equivalently $\langle \eta'(x) \rangle$)

choose $i \leq c+1$ such that $+d$ moves $\eta(x_i), \ldots, \eta(x_c)$ but not $\eta(x_1), \ldots, \eta(x_{i-1})$ from below to above some integer

choose d' that does the same for η'

Model-checking TCTL: proof

Assume $\mathbb{A}$ has a clock $x_{\text{real}} \in C$ not mentioned in guards, invariants or φ.

For $r \in \mathbb{R}_{\geq 0}$ write $\langle r \rangle := r - \lfloor r \rfloor$. E.g., $\langle 1.23 \rangle = 0.23$.

$\eta \approx \eta'$ iff

- for all $x \in C$: $\lfloor \eta(x) \rfloor$ and $\lfloor \eta'(x) \rfloor$ are equal or both $\geq k$.
- for all $x, y \in C$ with $\eta(x) < k, \eta(y) < k$: $\langle \eta(x) \rangle \leq \langle \eta(y) \rangle$ iff $\langle \eta'(x) \rangle \leq \langle \eta'(y) \rangle$; $\langle \eta(x) \rangle = 0$ iff $\langle \eta'(x) \rangle = 0$

Claim: if $\eta \approx \eta'$ and $d \in \mathbb{R}_{\geq 0}$, then $\eta + d \approx \eta' + d'$ for some $d' \in \mathbb{R}_{\geq 0}$

Get: for $\eta_0 \approx \eta_0'$ and a relevant path

$$\langle \ell_0, \eta_0 \rangle \xrightarrow{d_0} \langle \ell_0, \eta_0 + d_0 \rangle \xrightarrow{\alpha_0} \langle \ell_1, \eta_1 \rangle \xrightarrow{d_1} \langle \ell_1, \eta_1 + d_1 \rangle \xrightarrow{\alpha_1} \langle \ell_2, \eta_2 \rangle \xrightarrow{d_2} \cdots$$

there are $d_i' \in \mathbb{R}_{\geq 0}$ and $\eta_i' \approx \eta_i$ such that

$$\langle \ell_0, \eta_0' \rangle \xrightarrow{d_0'} \langle \ell_0, \eta_0' + d_0' \rangle \xrightarrow{\alpha_0} \langle \ell_1, \eta_1' \rangle \xrightarrow{d_1'} \langle \ell_1, \eta_1' + d_1' \rangle \xrightarrow{\alpha_1} \langle \ell_2, \eta_2' \rangle \xrightarrow{d_2'} \cdots$$

is a relevant path (by the proof of the claim).

Model-checking TCTL: proof

Assume $\mathbb{A}$ has a clock $x_{\text{real}} \in C$ not mentioned in guards, invariants or φ.

For $r \in \mathbb{R}_{\geq 0}$ write $\langle r \rangle := r - \lfloor r \rfloor$. E.g., $\langle 1.23 \rangle = 0.23$.

$\eta \approx \eta'$ iff

- for all $x \in C$: $\lfloor \eta(x) \rfloor$ and $\lfloor \eta'(x) \rfloor$ are equal or both $\geq k$.
- for all $x, y \in C$ with $\eta(x) < k, \eta(y) < k$: $\begin{array}{l} \langle \eta(x) \rangle \leq \langle \eta(y) \rangle \text{ iff } \langle \eta'(x) \rangle \leq \langle \eta'(y) \rangle \\ \langle \eta(x) \rangle = 0 \text{ iff } \langle \eta'(x) \rangle = 0 \end{array}$

Claim: if $\eta \approx \eta'$ and $d \in \mathbb{R}_{\geq 0}$, then $\eta + d \approx \eta' + d'$ for some $d' \in \mathbb{R}_{\geq 0}$

Get: for $\eta_0 \approx \eta_0'$ and a relevant path

$$\langle \ell_0, \eta_0 \rangle \stackrel{d_0}{\rightarrow} \langle \ell_0, \eta_0 + d_0 \rangle \stackrel{\alpha_0}{\rightarrow} \langle \ell_1, \eta_1 \rangle \stackrel{d_1}{\rightarrow} \langle \ell_1, \eta_1 + d_1 \rangle \stackrel{\alpha_1}{\rightarrow} \langle \ell_2, \eta_2 \rangle \stackrel{d_2}{\rightarrow} \cdots$$

choose $d_i' \in \mathbb{R}_{\geq 0}$ accordingly so that

$$\langle \ell_0, \eta_0' \rangle \stackrel{d_0'}{\rightarrow} \langle \ell_0, \eta_0' + d_0' \rangle \stackrel{\alpha_0}{\rightarrow} \langle \ell_1, \eta_1' \rangle \stackrel{d_1'}{\rightarrow} \langle \ell_1, \eta_1' + d_1' \rangle \stackrel{\alpha_1}{\rightarrow} \langle \ell_2, \eta_2' \rangle \stackrel{d_2'}{\rightarrow} \cdots$$

is a relevant path (by the proof of the claim).

Then: $\quad \langle \ell, \eta \rangle \models \varphi \iff \langle \ell, \eta' \rangle \models \varphi$

Model-checking TCTL: proof

Region transition system $\mathbb{R}(\mathbb{A}, k)$

- states: $\langle \ell, [\eta] \rangle$ for a location ℓ and region $[\eta] := \{\eta' \mid \eta \approx \eta'\}$
- initial states: $\langle \ell_0, [\eta_0] \rangle$ for a initial location ℓ and η_0 constantly 0
- propositional variables: $AP \cup CC$
- label of $\langle \ell, [\eta] \rangle$ is the label of $\langle \ell, \eta \rangle$ in $TS(\mathbb{A})$
- actions: $Act \mathbin{\dot\cup} \{\tau\}$
- transitions:
  - $\langle \ell, [\eta] \rangle \xrightarrow{\alpha} \langle \ell', [\eta'] \rangle$ if $\langle \ell, \eta \rangle \xrightarrow{\alpha} \langle \ell', \eta' \rangle$ in $TS(\mathbb{A})$ and $\alpha \in Act$
  - $\langle \ell, [\eta] \rangle \xrightarrow{\tau} \langle \ell, [\eta'] \rangle$ if η' is the successor of η

    both $\approx \eta_\infty :=$ constantly k, or, $\eta \not\approx \eta' \approx \eta + d$ for some $d \in \mathbb{R}_{\geq 0}$ such that for all $d' < d$: $\eta + d' \in [\eta] \cup [\eta']$.

Size $k^{O(c)}$.

Model-checking TCTL: proof

Idea for each subformula ψ of φ compute

$$Ext(\psi) := \{\langle \ell, [\eta] \rangle \mid \langle \ell, \eta \rangle \models \psi \text{ and is reachable}\}$$

Case $\psi = \exists(\psi_0\ U_{[a,b)}\ \psi_1)$.

Assume we already computed $Ext(\psi_0)$ and $Ext(\psi_1)$.

Problem How to decide $\langle \ell, \eta \rangle \models \psi$? Wlog $\eta(x_{\mathsf{real}}) = 0$.

Model-checking TCTL: proof

Idea for each subformula ψ of φ compute

$$Ext(\psi) := \{\langle \ell, [\eta] \rangle \mid \langle \ell, \eta \rangle \models \psi \text{ and is reachable}\}$$

Case $\psi = \exists(\psi_0 \; U_{[a,b)} \; \psi_1)$.

Assume we already computed $Ext(\psi_0)$ and $Ext(\psi_1)$.

Problem How to decide $\langle \ell, \eta \rangle \models \psi$? Wlog $\eta(x_{\mathsf{real}}) = 0$.

Claim $\langle \ell, \eta \rangle$ reachable with $\eta(x_{\mathsf{real}}) = 0$. Then $\langle \ell, \eta \rangle \models \psi$ iff there is a path

$$\langle \ell, [\eta] \rangle = \langle \ell_0, [\eta_0] \rangle \; \langle \ell_1, [\eta_1] \rangle \cdots \langle \ell_n, [\eta_n] \rangle$$

for some $n \in \mathbb{N}$ such that

- $\langle \ell_i, [\eta_i] \rangle \in Ext(\psi_0) \cup Ext(\psi_1)$ for all $i < n$
- $\langle \ell_n, [\eta_n] \rangle \in Ext(\psi_1)$
- $a \leq \eta_n(x_{\mathsf{real}}) < b$

Then standard reachability algorithmics imply the theorem.

Model-checking TCTL: proof

$\Rightarrow$ Assume $\langle \ell, \eta \rangle \models \psi$, i.e., there is a relevant path π in $TS(\mathbb{A})$

$$\langle \ell, \eta \rangle = \langle \ell_0, \eta_0 \rangle \xrightarrow{d_0} \langle \ell_0, \eta_0 + d_0 \rangle \xrightarrow{\alpha_0} \langle \ell_1, \eta_1 \rangle \xrightarrow{d_1} \langle \ell_1, \eta_1 + d_1 \rangle \xrightarrow{\alpha_1} \langle \ell_2, \eta_2 \rangle \xrightarrow{d_2} \cdots$$

such that - $\langle \ell^t, \eta^t \rangle$ at some time $\eta^t(x_{\mathsf{real}}) = t \in [a, b)$ in π satisfies ψ_1

- all $\langle \ell', \eta' \rangle$ before $\langle \ell^t, \eta^t \rangle$ satisfy $\psi_0 \vee \psi_1$.

Model-checking TCTL: proof

$\Rightarrow$ Assume $\langle \ell, \eta \rangle \models \psi$, i.e., there is a relevant path π in $TS(\mathbb{A})$

$$\langle \ell, \eta \rangle = \langle \ell_0, \eta_0 \rangle \xrightarrow{d_0} \langle \ell_0, \eta_0 + d_0 \rangle \xrightarrow{\alpha_0} \langle \ell_1, \eta_1 \rangle \xrightarrow{d_1} \langle \ell_1, \eta_1 + d_1 \rangle \xrightarrow{\alpha_1} \langle \ell_2, \eta_2 \rangle \xrightarrow{d_2} \cdots$$

such that - $\langle \ell^t, \eta^t \rangle$ at some time $\eta^t(x_{\mathsf{real}}) = t \in [a, b)$ in π satisfies ψ_1

- all $\langle \ell', \eta' \rangle$ before $\langle \ell^t, \eta^t \rangle$ satisfy $\psi_0 \vee \psi_1$.

Choose $i \in \mathbb{N}$ and $d \in [0, d_i]$ such that

$$\langle \ell_0, \eta_0 \rangle \xrightarrow{d_0} \cdots \xrightarrow{\alpha_i} \langle \ell_i, \eta_i \rangle \xrightarrow{d} \langle \ell^t, \eta^t \rangle$$

This gives a finite path in $\mathbb{R}(\mathbb{A}, k)$:

$$\langle \ell_0, [\eta_0] \rangle \xrightarrow{\tau}^* \cdots \xrightarrow{\alpha_i} \langle \ell_i, [\eta_i] \rangle \xrightarrow{\tau}^* \langle \ell^t, [\eta^t] \rangle$$

where $\xrightarrow{\tau}^*$ abbreviates finitely many $\xrightarrow{\tau}$.

Model-checking TCTL: proof

$\Rightarrow$ Assume $\langle \ell, \eta \rangle \models \psi$, i.e., there is a relevant path π in $TS(\mathbb{A})$

$$\langle \ell, \eta \rangle = \langle \ell_0, \eta_0 \rangle \xrightarrow{d_0} \langle \ell_0, \eta_0 + d_0 \rangle \xrightarrow{\alpha_0} \langle \ell_1, \eta_1 \rangle \xrightarrow{d_1} \langle \ell_1, \eta_1 + d_1 \rangle \xrightarrow{\alpha_1} \langle \ell_2, \eta_2 \rangle \xrightarrow{d_2} \cdots$$

such that

- $\langle \ell^t, \eta^t \rangle$ at some time $\eta^t(x_{\text{real}}) = t \in [a, b)$ in π satisfies ψ_1
- all $\langle \ell', \eta' \rangle$ before $\langle \ell^t, \eta^t \rangle$ satisfy $\psi_0 \vee \psi_1$.

Choose $i \in \mathbb{N}$ and $d \in [0, d_i]$ such that

$$\langle \ell_0, \eta_0 \rangle \xrightarrow{d_0} \cdots \xrightarrow{\alpha_i} \langle \ell_i, \eta_i \rangle \xrightarrow{d} \langle \ell^t, \eta^t \rangle$$

This gives a finite path in $\mathbb{R}(\mathbb{A}, k)$:

$$\langle \ell_0, [\eta_0] \rangle \xrightarrow{\tau}^* \cdots \xrightarrow{\alpha_i} \langle \ell_i, [\eta_i] \rangle \xrightarrow{\tau}^* \langle \ell^t, [\eta^t] \rangle$$

where $\xrightarrow{\tau}^*$ abbreviates finitely many $\xrightarrow{\tau}$. Then

- $\langle \ell^t, [\eta^t] \rangle \in Ext(\psi_1)$
- for every other appearing $\langle \ell, [\eta] \rangle$ there is $\eta' \approx \eta$ such that $\langle \ell, \eta' \rangle$ is before $\langle \ell^t, \eta^t \rangle$ in π

  hence $\langle \ell, [\eta] \rangle \in Ext(\psi_0) \cup Ext(\psi_1)$

Model-checking TCTL: proof

$\Leftarrow$ Given an as-described path in $\mathbb{R}(\mathbb{A}, k)$

$$\langle \ell_0, [\eta_0] \rangle \ \langle \ell_1, [\eta_1] \rangle \cdots \langle \ell_n, [\eta_n] \rangle$$

Replace $\xrightarrow{\tau}$ by suitable $\xrightarrow{d}$ and get for suitable $\eta_i' \approx \eta_i$ a path in $TS(\mathbb{A})$

$$\langle \ell_0, \eta_0 \rangle \ \langle \ell_1, \eta_1' \rangle \cdots \langle \ell_n, \eta_n' \rangle$$

Make $\xrightarrow{d}$ / $\xrightarrow{\alpha}$ alternating by contracting consecutive $\xrightarrow{d}$ and adding $\xrightarrow{0}$ between consecutive $\xrightarrow{\alpha}$. Continue to a relevant path π (timelock-free).

Model-checking TCTL: proof

$\Leftarrow$ Given an as-described path in $\mathbb{R}(\mathbb{A}, k)$

$$\langle \ell_0, [\eta_0] \rangle \ \langle \ell_1, [\eta_1] \rangle \cdots \langle \ell_n, [\eta_n] \rangle$$

Replace $\xrightarrow{\tau}$ by suitable $\xrightarrow{d}$ and get for suitable $\eta'_i \approx \eta_i$ a path in $TS(\mathbb{A})$

$$\langle \ell_0, \eta_0 \rangle \ \langle \ell_1, \eta'_1 \rangle \cdots \langle \ell_n, \eta'_n \rangle$$

Make $\xrightarrow{d}$ / $\xrightarrow{\alpha}$ alternating by contracting consecutive $\xrightarrow{d}$ and adding $\xrightarrow{0}$ between consecutive $\xrightarrow{\alpha}$. Continue to a relevant path π (timelock-free).

Then

- $\langle \ell_n, \eta'_n \rangle$ is at time $\eta'_n(x_{\mathsf{real}}) \in [a, b)$ in π and satisfies ψ_1.
- for every $\langle \ell', \eta' \rangle$ before $\langle \ell_n, \eta'_n \rangle$ there is $i \leq n$ st $\ell' = \ell_i$ and $\eta' \approx \eta_i$, hence $\langle \ell', \eta' \rangle \models \psi_0 \vee \psi_1$.

Thus $\langle \ell, \eta \rangle \models \exists\big(\psi_0 \ U_{[a,b)} \ \psi_1\big)$. $\square$