Bernd Finkbeiner
Sven Schewe

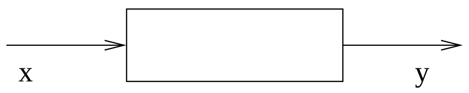
Date: April 17, 2008

Automata, Games and Verification: Lecture 1

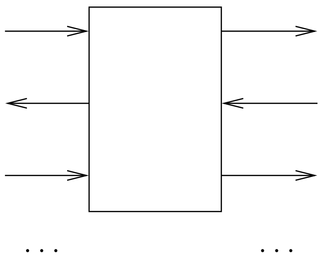
1 Motivation

We distinguish

- Transformational programs



- Reactive systems



  - nonterminating behavior
  - interaction (program vs. environment)

1.1 Problem 1: Verification

Example: Mutual execution with program TURN

local t: boolean where initially $t = 0$

$$P_0 :: \left[\begin{array}{l} \text{loop forever do} \\ \qquad \left[\begin{array}{ll} 00: & \text{noncritical;} \\ 01: & \text{await } t = 0; \\ 10: & \text{critical;} \\ 11: & t := 1; \end{array}\right] \end{array}\right] \;||\; P_1 :: \left[\begin{array}{l} \text{loop forever do} \\ \qquad \left[\begin{array}{ll} 00: & \text{noncritical;} \\ 01: & \text{await } t = 1; \\ 10: & \text{critical;} \\ 11: & t := 0; \end{array}\right] \end{array}\right]$$

TURN is a finite-state program with 32 states, which can be encoded as bit vectors $(b_1, b_2, b_3, b_4, b_5)$, with (b_1, b_2) for the location of P_0, (b_3, b_4) for the location of P_1, and b_5 for t. ✚

Behavior: infinite sequence of states
Specification: set of correct behaviors

Example: specifications:

- Mutual execution: it is never the case that P_0 and P_1 are in their critical sections, i.e. the states 10100 and 10101 do not occur
- Accessibility: whenever P_i is in location 01 it will eventually reach location 10

✚

The Verification Problem: Given a program P and a specification φ, decide whether P satisfies φ.

Underlying concept: Automata over infinite words (more generally: objects)

Solution:

1. Construct automaton that accepts all sequences that are
   - possible in P and
   - violate φ.
2. Check automaton for emptiness.

1.2 Problem 2: Synthesis

Example: Mutual execution by arbiter

local t, r_1, r_2: boolean where initially $t = r_1 = r_2 = 0$

$$P_0 :: \left[\begin{array}{l} \text{loop forever do} \\ \quad \left[\begin{array}{ll} 00: & r_0 := 1; \\ 01: & \text{await } t = 0; \\ 10: & \text{critical;} \\ 11: & r_0 := 0; \end{array}\right] \end{array}\right] \;||\; P_1 :: \left[\begin{array}{l} \text{loop forever do} \\ \quad \left[\begin{array}{ll} 00: & r_1 := 1; \\ 01: & \text{await } t = 1; \\ 10: & \text{critical;} \\ 11: & r_1 := 0; \end{array}\right] \end{array}\right] \;||\; \text{Arbiter:: } \mathbf{?}$$

✚

The Synthesis Problem: Given a specification φ, decide if *there exists* a program P that satisfies φ. If yes: construct such a program.

Underlying concept: Infinite games.
Play of the game = infinite sequence of states.
Player "system" wins the game if sequence satisfies φ for all possible behaviors of player "environment".

Solution:

1. Decide whether player "system" has a winning strategy.
2. If yes, construct a program that implements that strategy.

1.3 History

1960 − 1970 Fundamental results about ω-automata and games. Motivation: Logical decision problems, circuit design.

- **J. Richard Büchi** (1924-1984)
  Swiss logician and mathematician; Ph.D. at ETH, then Purdue University, Lafayette, Indiana. Inventor of Büchi automata. Great influence on theoretical computer science, combinatorics, grapth theory.
- **Robert McNaughton**
  taught philosophy; then switched to computer science in 1950s; emeritus at Harvard; McNaughton's theorem: each recognizable set of infinite words can be recognized by a deterministic ω-automaton.
- **Michael Rabin** (*1931, Breslau)
  won Turing award together with Dana Scott for inventing nondeterministic machines; proved that second order theory of n successors is decidable; determinacy of parity games.

Since 1980: Revival of the theory in the setting of temporal logics

Motivation today:

- industrial use (especially finite-state verification "model checking")
- decidability of many problems with infinite structures
- bridge between logic and computer science

2 Büchi Automata

2.1 Basic Definitions

- The *set of natural numbers* $\{0, 1, 2, 3, \ldots\}$ is denoted by ω.
- An *alphabet* Σ is a finite set of symbols.
- An *infinite sequence/string/word* is a function from natural numbers to an alphabet:
  $\alpha : \omega \to \Sigma$
  An infinite word is composed of its letters, so that in particular $\alpha = \alpha(0)\alpha(1)\alpha(2)\ldots$
- The *set of infinite words over alphabet* Σ is denoted Σ^ω (finite words: Σ^*).
- An *ω-language* L is a subset of Σ^ω.

Example:

- $\emptyset$ is the *empty* ω-language.

- $\{a^\omega\} = \{aaaa\ldots\}$;
- $\{ba^\omega, aba^\omega, aaba^\omega, \ldots\}$.

✚

Definition 1 *A* nondeterministic Büchi automaton $\mathcal{A}$ *over alphabet* Σ *is a tuple* (S, I, T, F)*:*

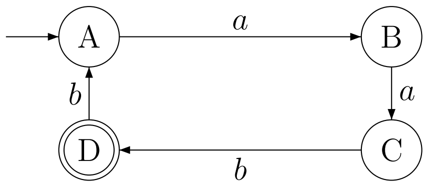
- S *: a finite set of* states
- $I \subseteq S$ *: a subset of* initial states
- $T \subseteq S \times \Sigma \times S$ *: a set of* transitions
- $F \subseteq S$ *: a subset of* accepting/final states

Now we define how a Büchi automaton uses an infinite word as input. Notice that we do not refer to acceptance in this definition.

Definition 2 *A* run *of a nondeterministic Büchi automaton* $\mathcal{A}$ *on an infinite input word* $\alpha = \sigma_0\sigma_1\sigma_2\ldots$ *is an infinite sequence of states* $s_0, s_1, s_2, \ldots$ *such that the following hold:*

- $s_0 \in I$
- *for all* $i \in \omega$, $(s_i, \sigma_i, s_{i+1}) \in T$

Example:



In the automaton shown the set of states are $S = \{A, B, C, D\}$, the initial set of states are $I = \{A\}$ (indicated with pointing arrow with no source), the transitions $T = \{(A, a, B), (B, a, C), (C, b, D), (D, b, A)\}$ are the remaining arrows in the diagram, and the set of accepting states is $F = \{D\}$ (double-lined state circle).

On input $aabbaabb\ldots$ the Büchi automaton shown has only the run:

$ABCDABCDABCD\ldots$ ✚

Determinism is a property of machines that can only react in a unique way to their input. The following definition makes this clear for Büchi automata.

Definition 3 *A Büchi automaton $\mathcal{A}$ is* deterministic *when T is a partial function (with respect to the next input letter and the current state):*
$\forall \sigma \in \Sigma, \forall s, s_0, s_1 \in S \,.\, (s, \sigma, s_0) \in T$ *and* $(s, \sigma, s_1) \in T \;\Rightarrow\; s_0 = s_1$
and I is a singleton.

(By Büchi automaton we usually mean nondeterministic Büchi automaton.)

Definition 4 *The* infinity set *of an infinite word $\alpha \in \Sigma^\omega$ is the set* $In(\alpha) = \{\sigma \in \Sigma \,|\, \forall i \exists j \,.\, j \geq i$ *and* $\alpha(j) = \sigma\}$

Definition 5

- *A Büchi automaton $\mathcal{A}$* accepts *an infinite word α if:*
  - *there is a run $r = s_0 s_1 s_2 \ldots$ of α on $\mathcal{A}$*
  - *r is* accepting*: $In(r) \cap F \neq \emptyset$*
- *The* language recognized by Büchi automaton $\mathcal{A}$ *is defined as follows:*
  $\mathcal{L}(\mathcal{A}) = \{\alpha \in \Sigma^\omega \,|\, \mathcal{A} \text{ accepts } \alpha\}$

Example: Automaton $\mathcal{A}$ from previous example. $\mathcal{L}(\mathcal{A}) = \{aabbaabbaabb\ldots\}$. ✚

Comment: A deterministic Büchi automaton $\mathcal{A} = (S, I, T, F)$ defines a partial function[1] from Σ^ω to a set of runs $R \subseteq S^\omega$. **End Comment**

Definition 6 *An ω-language L is Büchi recognizable if there is a Büchi automaton $\mathcal{A}$ such that $\mathcal{L}(\mathcal{A}) = L$.*

Example: The singleton ω-language $L = \{\sigma\}$ with $\sigma = abaabaaabaaaab\ldots$ is not Büchi recognizable. (Note that all finite languages of finite words are NFA-recognizable. Analog result does not hold for Büchi-automata)

Proof:

- Suppose there is a Büchi automaton $\mathcal{A}$ with $\mathcal{L}(\mathcal{A}) = L$.
- Let $r = s_0 s_1 \ldots$ be an accepting run on σ.
- Since F is finite, there exists $k, k' \in \omega$ with $k < k'$ and $s_k = s_{k'} \in F$.
- $r' = r_0 \ldots r_{k'-1}(r_k \ldots r_{k'-1})$ is an accepting run on $\sigma' = \sigma(0) \ldots \sigma(k'-1)(\sigma(k) \ldots \sigma(k'-1))^\omega$.
- Hence, $\sigma' \in \mathcal{L}(\mathcal{A})$. Contradiction.

■

✚

Definition 7 *A Büchi automaton is* complete *if its transition relation contains a function:*
$\forall s \in S \sigma \in \Sigma \exists s' \in S \,.\, (s, \sigma, s') \in T$

[1] A *partial function* is a function that is not defined on all of the elements of its domain.

Theorem 1 *For every Büchi automaton $\mathcal{A}$, there is a complete Büchi automaton $\mathcal{A}'$ such that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$.*

Proof:

We define $\mathcal{A}'$ in terms of the components S, I, T, F of $\mathcal{A}$:

$S' = S \cup \{f\} \qquad f$ new

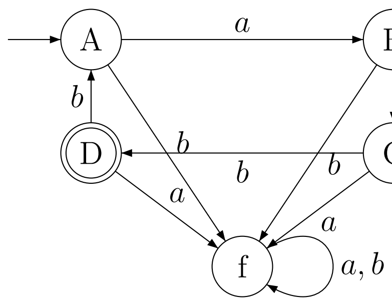
$I' = I$

$T' = T \cup \{(s, \sigma, f) \mid \nexists s' \,.\, (s, \sigma, s') \in T\} \cup \{(f, \sigma, f) \mid \sigma \in \Sigma\}$

$F' = F$

The runs of $\mathcal{A}'$ are a superset of those of $\mathcal{A}$ since we have added states and transistions. Furthermore, on any infinite input word α the accepting runs of $\mathcal{A}$ and $\mathcal{A}'$ correspond, because any run that reaches f stays in f, and since $f \notin F'$, such a run is not accepting. ■

Example: Completing the Büchi automaton from a previous example we obtain the following automaton:



✚

Unless we specify otherwise, we will only consider complete automata when we prove results.

Comment: A complete deterministic Büchi automaton $\mathcal{A} = (S, I, T, F)$ may be viewed as a total function[2] from Σ^ω to S^ω. A complete (possibly nondeterministic) Büchi automaton can produce at least one run for every Σ^ω input word.
End Comment

[2]A total function, in contrast to a partial one, is defined on its entire domain.

Bernd Finkbeiner
Sven Schewe

Date: April 24, 2008

Automata, Games and Verification: Lecture 2

3 ω-regular Languages

Definition 1 *The* ω-regular expressions *are defined as follows.*

- *If R is an regular expression where $\epsilon \notin \mathcal{L}(R)$,*
  then R^ω is an ω-regular expression.
  $\mathcal{L}(R^\omega) = \mathcal{L}(R)^\omega$
  where $L^\omega = \{u_0 u_1 \ldots \mid u_i \in L, |u_i| > 0$ for all $i \in \omega\}$ for $L \subseteq \Sigma^*$.
- *If R is a regular expression and U is an ω-regular expression,*
  then $R \cdot U$ is an ω-regular expression.
  $\mathcal{L}(R \cdot U) = \mathcal{L}(R) \cdot \mathcal{L}(U)$
  where $L_1 \cdot L_2 = \{r \cdot u \mid r \in L_1, u \in L_2\}$ for $L_1 \subseteq \Sigma^*, L_2 \subseteq \Sigma^\omega$.
- *If U_1 and U_2 are ω-regular expressions,*
  then $U_1 + U_2$ is an ω-regular expression.
  $\mathcal{L}(U_1 + U_2) = \mathcal{L}(U_1) \cup \mathcal{L}(U_2)$.

Definition 2 *An* ω-regular *language is a finite union of ω-languages of the form $U \cdot V^\omega$ where $U, V \subseteq \Sigma^*$ are regular languages.*

Theorem 1 *If L_1 and L_2 are Büchi recognizable, then so is $L_1 \cup L_2$.*

Proof:

Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be Büchi automata that recognize L_1 and L_2, respectively. We construct an automaton $\mathcal{A}'$ for $L_1 \cup L_2$:

- $S' = S_1 \cup S_2$ (w.l.o.g. we assume $S_1 \cap S_2 = \emptyset$);
- $I' = I_1 \cup I_2$;
- $T' = T_1 \cup T_2$;
- $F' = F_1 \cup F_2$.

$\mathcal{L}(\mathcal{A}') \subseteq \mathcal{L}(\mathcal{A}_1) \cup \mathcal{L}(\mathcal{A}_2)$: For $\alpha \in \mathcal{L}(A')$, we have an accepting run $r = s_0 s_1 s_2 \ldots$ of α in $\mathcal{A}'$. If $s_0 \in S_1$, then r is an accepting run on $\mathcal{A}_1$, otherwise $s_0 \in S_2$ and r is an accepting run on $\mathcal{A}_2$.

$\mathcal{L}(\mathcal{A}') \supseteq \mathcal{L}(\mathcal{A}_1) \cup \mathcal{L}(\mathcal{A}_2)$: For $i \in \{1, 2\}$ and $\alpha \in \mathcal{L}(\mathcal{A}_i)$, there is an accepting run $r = s_0 s_1 s_2 \ldots$ on $\mathcal{A}_i$. The run r is accepting for α in $\mathcal{A}'$. ■

Theorem 2 *If L_1 and L_2 are Büchi recognizable, then so is $L_1 \cap L_2$.*

Proof:

We construct an automaton $\mathcal{A}'$ from $\mathcal{A}_1$ and $\mathcal{A}_2$:

- $S' = S_1 \times S_2 \times \{1, 2\}$
- $I' = I_1 \times I_2 \times \{1\}$
- $T' = \{((s_1, s_2, 1), \sigma, (s_1', s_2', 1)) \mid (s_1, \sigma, s_1') \in T_1, (s_2, \sigma, s_2') \in T_2, s_1 \notin F_1\}$
  $\cup \{((s_1, s_2, 1), \sigma, (s_1', s_2', 2)) \mid (s_1, \sigma, s_1') \in T_1, (s_2, \sigma, s_2') \in T_2, s_1 \in F_1\}$
  $\cup \{((s_1, s_2, 2), \sigma, (s_1', s_2', 2)) \mid (s_1, \sigma, s_1') \in T_1, (s_2, \sigma, s_2') \in T_2, s_2 \notin F_2\}$
  $\cup \{((s_1, s_2, 2), \sigma, (s_1', s_2', 2)) \mid (s_1, \sigma, s_1') \in T_1, (s_2, \sigma, s_2') \in T_2, s_2 \in F_2\}$
- $F' = \{(s_1, s_2, 2) \mid s_1 \in S_1, s_2 \in F_2\}$

$\mathcal{L}(\mathcal{A}') = \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2)$:

- $r' = (s_1^0, s_2^0, t^0)(s_1^1, s_2^1, t^1)\ldots$ is a run of $\mathcal{A}'$ on input word σ iff $r_1 = s_1^0 s_1^1 \ldots$ is a run of $\mathcal{A}_1$ on σ and $r_2 = s_2^0 s_2^1 \ldots$ is a run of $\mathcal{A}_2$ on σ.
- r is accepting iff r_1 is accepting and r_2 is accepting.

■

Theorem 3 *If L_1 is a regular language and L_2 is Büchi recognizable, then $L_1 \cdot L_2$ is Büchi-recognizable.*

Proof:

Let $\mathcal{A}_1$ be a finite-word automaton that recognizes L_1 and $\mathcal{A}_2$ be a Büchi automaton that recognizes L_2. We construct:

- $S' = S_1 \cup S_2$ (w.l.o.g. we assume $S_1 \cap S_2 = \emptyset$;
- $I' = \begin{cases} I_1 & \text{if } I_1 \cap F_1 = \emptyset \\ I_1 \cup I_2 & \text{otherwise;} \end{cases}$
- $T' = T_1 \cup T_2 \cup \{(s, \sigma, s') \mid (s, \sigma, f) \in T_1, f \in F_1, s' \in I_2\}$;
- $F' = F_2$.

■

Theorem 4 *If L is a regular language then L^ω is Büchi recognizable.*

Proof:

Let $\mathcal{A}$ be a finite word automaton; let w.l.o.g. $\epsilon \notin \mathcal{L}(\mathcal{A})$.

- **Step 1:** Ensure that all initial states have no incoming transitions. We modify $\mathcal{A}$ as follows:
  - $S' = S \cup \{s_{\text{new}}\}$;
  - $I' = \{s_{\text{new}}\}$;
  - $T' = T \cup \{(s_{\text{new}}, \sigma, s') \mid (s, \sigma, s') \in T \text{ for some } s \in I\}$;

  - $F' = F$.

  This modification does not affect the language of $\mathcal{A}$.

- **Step 2:** Add loop:
  - $S'' = S'$; $I'' = I'$;
  - $T'' = T' \cup \{(s, \sigma, s_{\text{new}} \mid (s, \sigma, s') \in T'$ and $s' \in F'\}$;
  - $F'' = I'$.

$\mathcal{L}(\mathcal{A}'') \subseteq \mathcal{L}(\mathcal{A}')^\omega$:

- Assume $\alpha \in \mathcal{L}(\mathcal{A}'')$ and $s_0 s_1 s_2 \ldots$ is an accepting run for α in $\mathcal{A}''$.
- Hence, $s_i = s_{\text{new}} \in F'' = I'$ for infinitely many indices i: $i_0, i_1, i_2, \ldots$.
- This provides a series of runs in $\mathcal{A}'$:
  - run $s_0 s_1 \ldots s_{i_1 - 1} s$ on $w_1 = \alpha(0)\alpha(1) \ldots \alpha(i_1 - 1)$ for some $s \in F'$;
  - run $s_{i_1} s_{i_1+1} \ldots s_{i_2-1} s$ on $w_2 = \alpha(i_1)\alpha(i_1 + 1) \ldots \alpha(i_2 - 1)$ for some $s \in F'$;
  - $\ldots$
- This yields $w_k \in \mathcal{L}(\mathcal{A}')$ for every $k \geq 1$.
- Hence, $\alpha \in \mathcal{L}(\mathcal{A}')^\omega$.

$\mathcal{L}(\mathcal{A}'') \supseteq \mathcal{L}(\mathcal{A}')^\omega$:

- Let $\alpha = w_1 w_2 w_3 \in \Sigma^\omega$ such that $w_k \in \mathcal{L}(\mathcal{A}')$ for all $k \geq 1$.
- For each k, we choose an accepting run $s_0^k s_1^k s_2^k \ldots s_{n_k}^k$ of $\mathcal{A}'$ on w_k.
- Hence, $s_0^k \in I'$ and $s_{n_k}^k \in F'$ for all $k \geq 1$.
- Thus,
  $$s_0^1 \ldots s_{n_1-1}^1 s_0^2 \ldots s_{n_2-1}^2 s_0^3 \ldots s_{n_3-1}^3 \ldots$$
  is an accepting run on α in $\mathcal{A}''$.
- Hence, $\alpha \in \mathcal{L}(\mathcal{A}'')$.

■

Theorem 5 (Büchi's Characterization Theorem (1962)) *An ω-language is Büchi recognizable iff it is ω-regular.*

Proof:

"$\Leftarrow$" follows from previous constructions.

"$\Rightarrow$": Given a Büchi automaton $\mathcal{A}$, we consider for each pair $s, s' \in S$ the regular language

$$W_{s,s'} = \{u \in \Sigma^* \mid \text{finite-word automaton } (S, \{s\}, T, \{s'\}) \text{ accepts } u\ \} .$$

Claim: $\mathcal{L}(\mathcal{A}) = \bigcup_{s \in I, s' \in F} W_{s,s'} \cdot {W_{s',s'}}^\omega$.

$\mathcal{L}(\mathcal{A}) \subseteq \bigcup_{s \in I, s' \in F} W_{s,s'} \cdot {W_{s',s'}}^\omega$:

- Let $\alpha \in \mathcal{L}(\mathcal{A})$.
- Then there is an accepting run r for α on $\mathcal{A}$, which begins at some $s \in I$ and visits some $s' \in F$ infinitely often:

  $$r : s \xrightarrow{\alpha_0} s' \xrightarrow{\alpha_1} s' \xrightarrow{\alpha_2} s' \xrightarrow{\alpha_3} s' \xrightarrow{\alpha_4} s' \rightarrow \ \ldots,$$

  where $\alpha = \alpha_0 \cdot \alpha_1 \cdot \alpha_2 \cdot \alpha_3 \cdot \alpha_4 \cdot \ldots$.
  (Notation:
  $s_0 \overset{\sigma_0\sigma_1,\ldots\sigma_k}{\longrightarrow} s_{k+1}$: there exist $s_1, \ldots s_k$ s.t. $(s_i, \sigma_i, s_{i+1}) \in$ for all $0 \leq i \leq k$.)
- Hence, $\alpha_0 \in W_{s,s'}$ and $\alpha_k \in W_{s',s'}$ for $k > 0$ and thus $\alpha \in W_{s,s'} \cdot {W_{s',s'}}^{\omega}$ for some $s \in I, s' \in F$.

$\mathcal{L}(\mathcal{A}) \supseteq \bigcup_{s \in I, s' \in F} W_{s,s'} \cdot {W_{s',s'}}^{\omega}$:

- Let $\alpha = \alpha_0 \cdot \alpha_1 \cdot \alpha_2 \cdot \ldots$ with $\alpha_0 \in W_{s,s'}$ and $\alpha_k \in W_{s',s'}$ for some $s \in I, s' \in F$.
- Then the run

  $$r : s \xrightarrow{\alpha_0} s' \xrightarrow{\alpha_1} s' \xrightarrow{\alpha_2} s' \xrightarrow{\alpha_3} s' \xrightarrow{\alpha_4} s' \rightarrow$$

  exists and is accepting since $s' \in F$.
- It follows that $\alpha \in \mathcal{L}(\mathcal{A})$.

■

4 Deterministic Büchi Automata

Theorem 6 *The language $L = \{\alpha \in \Sigma^\omega \mid In(\alpha) = \{b\}\}$ over $\Sigma = \{a, b\}$ is not recognizable by a deterministic Büchi automaton.*

Proof:

- Assume that L is recognized by the deterministic Büchi automaton $\mathcal{A}$.
- Since $b^\omega \in L$, there is a run
  $r_0 = s_{0,0}s_{0,1}s_{0,2}, \ldots$
  with $s_{0,n_0} \in F$ for some $n_0 \in \omega$.
- Similarly, $b^{n_0}ab^\omega \in L$ and there must be a run
  $r_1 = s_{0,0}s_{0,1}s_{0,2} \ldots s_{0,n_0}s_1s_{1,0}s_{1,1}s_{1,2} \ldots$
  with $s_{1,n_1} \in F$
- Repeating this argument, there is a word
  $b^{n_0}ab^{n_1}ab^{n_2}a \ldots$
  accepted by $\mathcal{A}$.
- This contradicts $L = \mathcal{L}(\mathcal{A})$.

■

Bernd Finkbeiner
Sven Schewe

Date: May 8, 2008

Automata, Games and Verification: Lecture 3

Definition 1 (Substrings) *Let $\alpha \in \Sigma^*$. For two integers $n \leq m$ we define*

$$\alpha(n,m) = \alpha(n)\alpha(n+1)\ldots\alpha(m) \ .$$

Definition 2 (Limit) *For $W \subseteq \Sigma^*$:*

$$\overrightarrow{W} = \{\alpha \in \Sigma^\omega \mid \textit{there exist infinitely many } n \in \omega \textit{ s.t. } \alpha(0,n) \in W\} \ .$$

Theorem 1 *An ω-language $L \subseteq \Sigma^\omega$ is recognizable by a deterministic Büchi automaton iff there is a regular language $W \subseteq \Sigma^*$ s.t. $L = \overrightarrow{W}$.*

Proof:

Let L be the language of a deterministic Büchi automaton $\mathcal{A}$; let W be the regular language of $\mathcal{A}$ as a deterministic finite-word automaton. We show that $L = \overrightarrow{W}$.

$$\begin{array}{ll} & \alpha \in L \\ \text{iff} & \text{for the unique run } r \text{ of } \mathcal{A} \text{ on } \alpha,\ \mathit{In}(r) \cap F \neq \emptyset \\ \text{iff} & \alpha(0,n) \in W \text{ for infinitely many } n \in \omega \\ \text{iff} & \alpha \in \overrightarrow{W}. \end{array}$$

■

5 Complementation

Theorem 2 *For any deterministic Büchi automaton $\mathcal{A}$, there exists a Büchi automaton $\mathcal{A}'$ such that $\mathcal{L}(\mathcal{A}') = \Sigma^\omega \setminus \mathcal{L}(\mathcal{A})$.*

Proof:

We construct $\mathcal{A}'$ as follows:

- $S' = (S \times \{0\}) \cup ((S \setminus F) \times \{1\})$.
- $I' = I \times \{0\}$.
- $T' = \{((s,0),\sigma,(s',0)) \mid (s,\sigma,s') \in T\} \cup \{((s,0),\sigma,(s',1)) \mid (s,\sigma,s') \in T\} \cup \{((s,1),\sigma,(s,1)) \mid (s,\sigma,s') \in T, s' \in S - F\}$.
- $F' = (S - F) \times \{1\}$.

$\mathcal{L}(\mathcal{A}') \subseteq \Sigma^\omega - \mathcal{L}(\mathcal{A})$:

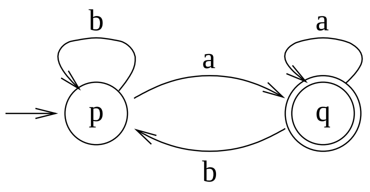
- For $\alpha \in \mathcal{L}(\mathcal{A}')$ we have an accepting run
$$r' : (s_0, 0)(s_1, 0) \dots (s_j, 0)(s'_0, 1)(s'_1, 1) \dots$$
on $\mathcal{A}'$.
- Hence,
$$r : s_0 s_1 s_2 \dots s_j s'_0 s'_1 \dots$$
is the unique run on α in $\mathcal{A}$.
- Since $s'_0, s'_1, \dots \in S \setminus F$, $In(r) \subseteq S \setminus F$. Hence, r is not accepting and $\alpha \in \Sigma^\omega - \mathcal{L}(\mathcal{A})$

$\mathcal{L}(\mathcal{A}') \supseteq \Sigma^\omega - \mathcal{L}(\mathcal{A})$:

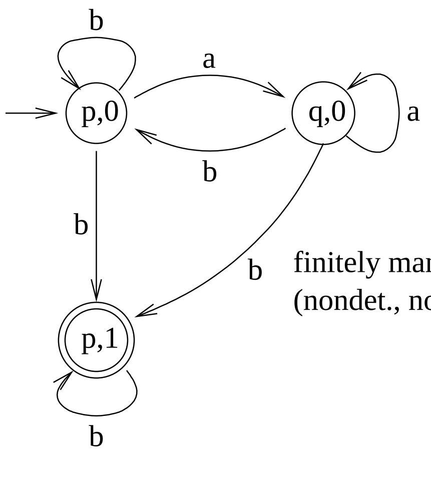
- We assume $\alpha \notin \mathcal{L}(\mathcal{A})$. Since $\mathcal{A}$ is deterministic and complete there exists a run
$$r : s_0 s_1 s_2 \dots$$
for α on $\mathcal{A}$, but $In(r) \cap F = \emptyset$.
- Thus there exists a $k \in \omega$ such that $s_j \notin F$ for $j > k$.
- This gives us the run
$$r' : (s_0, 0)(s_1, 0) \dots (s_k, 0)(s_{k+1}, 1)(s_{k+2}, 1) \dots$$
for α on $\mathcal{A}'$ with the property $In(r') \subseteq ((S - F) \times \{1\}) = F'$.
- Hence, r' is accepting and $\alpha \in \mathcal{L}(\mathcal{A}')$.

■

Example:



infinitely many 'a's



finitely many 'a's
(nondet., not complete)

✚

Reference: The following construction for the complementation of nondeterministic Büchi automata is taken from: Orna Kupferman and Moshe Y. Vardi, Weak alternating automata are not that weak. *ACM Trans. Comput. Logic* 2, 3 (Jul. 2001), 408-429.

Definition 3 *Let $\mathcal{A} = (S, I, T, F)$ be a nondeterministic Büchi automaton. The* run DAG *of $\mathcal{A}$ on a word $\alpha \in \Sigma^\omega$ is the directed acyclic graph $G = (V, E)$ where*

- $V = \bigcup_{l \geq 0}(S_l \times \{l\})$ *where $S_0 = I$ and* $s_{l+1} = \bigcup_{s \in S_l, (s,\alpha(l),s') \in T}\{s'\}$
- $E = \{(\langle s, l\rangle, \langle s', l+1\rangle) \mid l \geq 0, (s, \alpha(l), s') \in T\}$

A path in a run DAG is accepting iff it visits F infinitely often. The automaton accepts α if some path is accepting.

Definition 4 *A* ranking *for G is a function $f : V \to \{0, \ldots, 2 \cdot |S|\}$ such that*

- *for all $\langle s, l\rangle \in V$, if $f(\langle s, l\rangle)$ is odd then $s \notin F$;*
- *for all $(\langle s, l\rangle, \langle s', l'\rangle) \in E$, $f(\langle s', l'\rangle) \leq f(\langle s, l\rangle)$.*

A ranking is *odd* iff for all paths $\langle s_0, l_0\rangle, \langle s_1, l_1\rangle, \langle s_2, l_2\rangle, \ldots$ in G, there is a $i \geq 0$ such that $f(\langle s_i, l_i\rangle)$ is odd and, for all $j \geq 0$, $f(\langle s_{i+j}, l_{i+j}\rangle) = f(\langle s_i, l_i\rangle)$.

Lemma 1 *If there exists an odd ranking for G, then $\mathcal{A}$ does not accept α.*

Proof:

- In an odd ranking, every path eventually gets trapped in a some odd rank.
- If $f(\langle s, l\rangle)$ is odd, then $s \notin F$.
- Hence, every path visits F only finitely often.

■

Let G' be a subgraph of G. We call a vertex $\langle s, l\rangle$

- *safe* in G' if for all vertices $\langle s', l'\rangle$ reachable from $\langle s, l\rangle$, $s' \notin F$, and
- *endangered* in G' if only finitely many vertices are reachable.

We define an infinite sequence $G_0 \supseteq G_1 \supseteq G_2 \supseteq \ldots$ of DAGs inductively as follows:

- $G_0 = G$
- $G_{2i+1} = G_{2i} \setminus \{\langle s, l\rangle \mid \langle s, l\rangle$ is endangered in $G_{2i}\}$
- $G_{2i+2} = G_{2i+1} \setminus \{\langle s, l\rangle \mid \langle s, l\rangle$ is safe in $G_{2i}\}$.

Lemma 2 *If $\mathcal{A}$ does not accept α, then the following holds: For every $i \geq 0$ there exists an l_i such that for all $j \geq l_i$ at most $|S| - i$ vertices of the form $\langle \_, j\rangle$ are in G_{2i}.*

Proof:

Proof by induction on i:

- $i = 0$: In G, for every l, there are at most $|S|$ vertices of the form $\langle \_, l\rangle$.
- $i \rightarrow i+1$:
  - Case G_{2i} is finite: then $G_{2(i+1)}$ is empty.
  - Case G_{2i} is infinite:
    - There must exist a safe vertex $\langle s, l\rangle$ in G_{2i+1}. (Otherwise, we can construct a path in G with infinitely many visits to F).
    - We choose $l_{i+1} = l$.
    - We prove that for all $j \geq l$, there are at most $|S| - (i+1)$ vertices of the form $\langle \_, j\rangle$ in G_{2i+2}.
      - Since $\langle s, l\rangle \in G_{2i+1}$, it is not endangered in G_{2i}.
      - Hence, there are infinitely many vertices reachable from $\langle s, l\rangle$ in G_{2i}.
      - By König's Lemma, there exists an infinite path $p = \langle s, l\rangle, \langle s_1, l+1\rangle, \langle s, l+2\rangle, \ldots$ in G_{2i}.
      - No vertex on p is endangered (there is an infinite path). Therefore, p is in G_{2i+1}.
      - All vertices on p are safe ($\langle s, l\rangle$ is safe) in G_{2i+1}. Therefore, none of the vertices on p are in G_{2i+2}.
      - Hence, for all $j \geq l$, the number of vertices of the form $\langle \_, l\rangle$ is strictly smaller than their number in G_{2i}.

■

Lemma 3 *If $\mathcal{A}$ does not accept α, then there exists an odd ranking for G.*

Proof:

- We define $f(\langle s, l\rangle) = 2i$ if $\langle s, l\rangle$ is endangered in G_{2i} and
- $f(\langle s, l\rangle) = 2i+1$ if $\langle s, l\rangle$ is safe in G_{2i}.
- f is a ranking:
  - by Lemma 2, G_j is empty for $j > 2 \cdot |S|$. Hence, $f : V \rightarrow \{0, \ldots, 2 \cdot |S|\}$.
  - if $\langle s', l'\rangle$ is a successor of $\langle s, l\rangle$, then $f(\langle s', l'\rangle) \leq f(\langle s, l\rangle)$
    - Let $j := f(\langle s, l\rangle)$.
    - Case j is even: vertex $\langle s, l\rangle$ is endangered in G_j; hence either $\langle s', l'\rangle$ is not in G_j, and therefore $f(\langle s, l\rangle) < j$; or $\langle s', l'\rangle$ is in G_j and endangered; hence, $f(\langle s, l\rangle) = j$.
    - Case j is odd: vertex $\langle s, l\rangle$ is safe in G_j; hence either $\langle s', l'\rangle$ is not in G_j, and therefore $f(\langle s, l\rangle) < j$; or $\langle s', l'\rangle$ is in G_j and safe; hence, $f(\langle s, l\rangle) = j$.
  - f is an odd ranking:
    - For every path $\langle s_0, l_0\rangle, \langle s_1, l_1\rangle, \langle s_2, l_2\rangle, \ldots$ in G there exists an $i \geq 0$ such that for all $j \geq 0$, $f(\langle s_{i+j}, l_{i+j}\rangle) = f(\langle s_i, l_i\rangle)$.

- Suppose that $k := f(\langle s_i, l_i \rangle)$ is even. Thus, $\langle s_i, l_i \rangle$ is endangered in G_k.
- Since $f(\langle s_{i+j}, l_{i+j} \rangle) = k$ for all $j \geq 0$, all $\langle s_{i+j}, l_{i+j} \rangle$ are in G_k.
- This contradicts that $\langle s_i, l_i \rangle$ is endangered in G_k.

$\blacksquare$

Bernd Finkbeiner
Sven Schewe

Date: May 15, 2008

Automata, Games and Verification: Lecture 4

Theorem 1 *For each Büchi automaton $\mathcal{A}$ there exists a Büchi automaton $\mathcal{A}'$ such that $\mathcal{L}(\mathcal{A}') = \Sigma^\omega \smallsetminus \mathcal{L}(\mathcal{A})$.*

Helpful definitions:

- A *level ranking* is a function $g : S \to \{0, \dots, 2 \cdot |S|\} \cup \{\bot\}$ such that if $g(s)$ is odd, then $s \notin F$.
- Let $\mathcal{R}$ be the set of all level rankings.
- A level ranking g' *covers* a level ranking g if, for all $s, s' \in S$, if $g(s) \geq 0$ and $(s, \sigma, s') \in T$, then $0 \leq g'(s') \leq g(s)$.

Proof:

We define $\mathcal{A}' = (S', I', T', F')$ with

- $S' = \mathcal{R} \times 2^S$;
- $I' = \{\langle g_0, \emptyset\rangle$, where $g_0(s) = 2 \cdot |S|$ if $s \in I$ and $g_0(s) = \bot$ if $s \notin I$;
- $T = \{(\langle g, \emptyset\rangle, \sigma, \langle g', P'\rangle) \mid g' \text{ covers } g, \text{ and } P' = \{s' \in S \mid g'(s') \text{ is even }\}$
  $\cup\ \{(\langle g, P\rangle, \sigma, \langle g', P'\rangle) \mid P \neq \emptyset, g' \text{ covers } g, \text{ and}$
  $P' = \{s' \in S \mid (s, \sigma, s') \in T, s \in P, g'(s') \text{ is even }\}$;
- $F = \mathcal{R} \times \{\emptyset\}$.

(Intuition: $\mathcal{A}'$ guesses the level rankings for the run DAG. The P component tracks the states whose corresponding vertices in the run DAG have even ranks. Paths that traverse such vertices should eventually reach a vertex with odd rank. The acceptance condition ensures that all paths visit a vertex with odd rank infinitely often.)

$\mathcal{L}(\mathcal{A}') \subseteq \Sigma^\omega \smallsetminus \mathcal{L}(\mathcal{A})$:

- Let $\alpha \in \mathcal{L}(\mathcal{A}')$ and let $r' = (g_0, P_0), (g_1, P_1), \dots$ be an accepting run of $\mathcal{A}'$ on α.
- Let $G = (V, E)$ be the run DAG of $\mathcal{A}$ on α.
- The function $f : \langle s, l\rangle \mapsto g_l(s), s \in S_l, l \in \omega$ is a ranking for G:
  - if $g_i(s)$ is odd then $s \notin F$;
  - for all $(\langle s, l\rangle, \langle s', l+1\rangle) \in E$, $g_{l+1}(s') \leq g_l(s)$.
- f is an odd ranking:

  - Assume otherwise. Then there exists a path $\langle s_0, l_0\rangle, \langle s_1, l_1\rangle, \langle s_2, l_2\rangle, \ldots$ in G such that for infinitely many $i \in \omega$, $f(\langle s_i, l_i\rangle)$ is even.
  - Hence, there exists an index $j \in \omega$, such that $f(\langle s_j, l_j\rangle)$ is even and, for all $k \geq 0$, $f(\langle s_{j+k}, l_{j+k}\rangle) = f(\langle s_j, l_j\rangle)$.
  - Since r' is accepting, $P_{j'} = \emptyset$ for infinitely many j'. Let j' be the smallest such index $\geq j$.
  - $P_{j'+1+k} \neq \emptyset$ for all $k \geq 0$.
  - Contradiction.
- Since there exists an odd ranking, $\alpha \notin \mathcal{L}(\mathcal{A})$.

$\Sigma^\omega \setminus \mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{A}')$:

- Let $\alpha \in \Sigma^\omega \setminus \mathcal{L}(\mathcal{A})$ and let $G = (V, E)$ be the run DAG of $\mathcal{A}$ on α.
- There exists an odd ranking f on G.
- There is a run $r' = (g_0, P_0), (g_1, P_1), \ldots$ of $\mathcal{A}'$ on α, where
  $$g_l(s) = \begin{cases} f(\langle s, l\rangle) & \text{if } s \in S_l; \\ \perp & \text{otherwise;} \end{cases}$$
  $P_0 = \emptyset$,
  $$P_{l+1} = \begin{cases} \{s \in S \mid g_{l+1}(s) \text{ is even }\} & \text{if } P_l = \emptyset, \\ \{s' \in S \mid \exists s \in S_l \cap P_l \,.\, (\langle s, l\rangle, \langle s', l+1\rangle) \in E, g_{l+1}(s') \text{ is even}\} & \text{otherwise.} \end{cases}$$
- r' is accepting. (Assume there is an index i such that $P_j \neq \emptyset$ for all $j \geq i$. Then there exists a path in G that visits an even rank infinitely often.)
- Hence, $\alpha \in \mathcal{L}(\mathcal{A}')$.

■

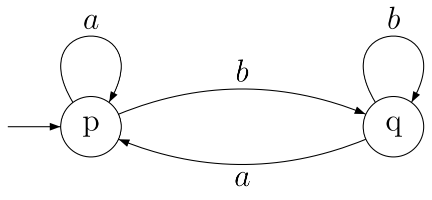
6 Muller Automata

Definition 1 *A* (nondeterministic) Muller automaton $\mathcal{A}$ *over alphabet* Σ *is a tuple* (S, I, T, F)*:*

- S, I, T *: defined as before*
- $\mathcal{F} \subseteq 2^S$ *: set of* accepting subsets*, called the* table.

Definition 2 *A run* r *of a Muller automaton is* accepting *iff* $In(r) \in F$

Example:



- for $\mathcal{F} = \{\{q\}\}$: $\mathcal{L}(\mathcal{A}) = (a \cup b)^* b^\omega$
- for $\mathcal{F} = \{\{q\}, \{p, q\}\}$: $\mathcal{L}(\mathcal{A}) = (a^* b)^\omega$

✚

Theorem 2 *For every (deterministic) Büchi automaton $\mathcal{A}$, there is (deterministic) Muller automaton $\mathcal{A}'$, such that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$.*

Proof:

$S' = S$, $I' = I$, $T' = T$
$\mathcal{F}' = \{Q \subseteq S \mid Q \cap F \neq \emptyset\}$ ■

Theorem 3 *For every nondeterministic Muller automaton $\mathcal{A}$ there is a nondeterministic Büchi automaton $\mathcal{A}'$ such that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$.*

Proof:

- $\mathcal{F} = \{F_1, \ldots, F_n\}$
- $S' = S \cup \bigcup_{i=1}^{n} \{i\} \times F_i \times 2^{F_i}$
- $I' = I$
- $T' = T$
  $\cup\ \{(s, \sigma, (i, s', \emptyset)) | 1 \leq i \leq n, (s, \sigma, s') \in T, s' \in F_i\}$
  $\cup\ \{((i, s, R), \sigma, (i', s', R')) \mid 1 \leq i \leq n, s, s' \in F_i, R, R' \subseteq F_i,$
  $(s, \sigma, s') \in T, R' = R \cup \{s\}$ if $R \neq F_i$ and $R' = \emptyset$ if $R = F_i\}$
- $F' = \bigcup_{i=1}^{n} \{i\} \times F_i \times \{F_i\}$

■

Boolean language operations: complementation, union, intersection.

Theorem 4 *The languages recognizable by deterministic Muller automata are closed under boolean operations.*

Proof:

- $\mathcal{L}(\mathcal{A}') = \Sigma^\omega \setminus \mathcal{L}(\mathcal{A})$:
  - $S' = S, I' = I, T' = T, \mathcal{F}' = 2^S \setminus \mathcal{F}$
- $\mathcal{L}(\mathcal{A}') = \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2)$:
  - $S' = S_1 \times S_2, I' = I_1 \times I_2,$
  - $T' = \{((s_1, s_2), \sigma, (s_1', s_2')) \mid (s_1, \sigma, s_1') \in T_1, (s_2, \sigma, s_2') \in T_2\}$
  - $\mathcal{F}' = \{\{(p_1, q_1), \ldots, (p_n, q_n)\} \mid \{p_1, \ldots, p_n\} \in \mathcal{F}_1, \{q_1, \ldots, q_n\} \in \mathcal{F}_2\}$
- $\mathcal{L}(\mathcal{A}_1) \cup \mathcal{L}(\mathcal{A}_2) = \Sigma^\omega \setminus ((\Sigma^\omega \setminus \mathcal{L}(\mathcal{A}_1)) \cap (\Sigma^\omega \setminus \mathcal{L}(\mathcal{A}_2)))$.

■

Theorem 5 *A language $\mathcal{L}$ is recognizable by a deterministic Muller automaton iff $\mathcal{L}$ is a boolean combination of languages $\overrightarrow{W}$ where $W \subseteq \Sigma^*$ is regular.*

Proof:

($\Leftarrow$)

- If W is regular, then $\overrightarrow{W}$ is recognizable by a deterministic Büchi automaton;
- hence, $\overrightarrow{W}$ is recognizable by a deterministic Muller automaton;
- hence, the boolean combination $\mathcal{L}$ is recognizable by a deterministic Muller automaton.

($\Rightarrow$) *left as an exercise.*

$\blacksquare$

Bernd Finkbeiner
Sven Schewe

Date: May 29, 2008

Automata, Games and Verification: Lecture 5

7 McNaughton's Theorem

Theorem 1 (McNaughton's Theorem (1966)) *Every Büchi recognizable language is recognizable by a deterministic Muller automaton.*

Definition 1 *A Büchi automaton (S, I, T, F) is called* semi-deterministic *if $S = N \uplus D$ is a partition of S, $F \subseteq D$ and $(D, \{d\}, T, F)$ is deterministic for every $d \in D$.*

Lemma 1 *For every Büchi automaton $\mathcal{A}$ there exists a semi-deterministic Büchi automaton $\mathcal{A}'$ with $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$.*

Proof:

Given $\mathcal{A} = (S, I, T, F)$, we construct $\mathcal{A}' = (S', I', T', F')$:

- $S' = 2^S \uplus 2^S \times 2^S$;
- $I' = \{I\}$;
- $T' = \{(L, \sigma, L') \mid L' = pr_3(T \cap L \times \{\sigma\} \times S)\}$;
  $\cup\{(L, \sigma, (\{s'\}, \emptyset)) \mid \exists s \in L. (s, \sigma, s') \in T\}$
  $\cup\{((L_1, L_2), \sigma, (L_1', L_2')) \mid L_1 \neq L_2$
  $L_1' = pr_3(T \cap L_1 \times \{\sigma\} \times S)$,
  $L_2' = pr_3(T \cap L_1 \times \{\sigma\} \times F) \cup pr_3(T \cap L_2 \times \{\sigma\} \times S)\}$
  $\cup\{((L, L), \sigma, (L_1', L_2')) \mid L_1' = pr_3(T \cap L_1 \times \{\sigma\} \times S)$,
  $L_2' = pr_3(T \cap L_1 \times \{\sigma\} \times F)\}$
- $F' = \{(L, L) \mid L \neq \emptyset)\}$

<u>$\mathcal{L}(\mathcal{A}') \subseteq \mathcal{L}(\mathcal{A})$:</u>

- Let $\alpha \in \mathcal{L}(\mathcal{A}')$.
- Let $r' = P_0, P_1, \ldots, P_n, (L_0, L_0'), (L_1, L_1'), \ldots$ be an accepting run of $\mathcal{A}'$ on α.
- For every $s \in L_0$ there is a run prefix of $\mathcal{A}$ on $\alpha(0, n)$, $p_0, p_1, \ldots, p_n, s$ such that $p_j \in P_j$ and
- Let $i_0, i_1, \ldots$ be an infinite sequence of indices such that $i_0 = 0$, $L_{i_j} = L_{i_j}'$, $L_{i_j} \neq \emptyset$ for all $j \in \omega$.
- For every $j > 1$, and every $s' \in L_{i_j}$ there exists a state $s \in L_{i_{j-1}}$ and a sequence $s = s_{i_{j-1}}, s_{i_{j-1}+1}, \ldots, s_{i_j} = s'$ such that $(s_k, \alpha(k), s_{k+1}) \in T$ for all $k \in \{i_{j-1}, \ldots, i_{i_j - 1}\}$ and $s_k \in F$ for some $k \in \{i_{j-1} + 1, \ldots, i_{i_j}\}$.
  Let $predecessor(s', i_j) := s$,
  $run(s', i_0) = p_0, p_1, \ldots, p_n, s'$ where $L_0 = \{s'\}$, and
  $run(s', i_j) = s_{i_{j-1}+1}, s_{i_{j-1}+2}, \ldots, s_{i_j}$, for $j > 0$.

- Consider the following $\left(\bigcup_{j\in\omega} L_{i_j} \times \{j\}\right)$-labeled tree:
  - the root is labeled with $(s, 0)$, where $L_0 = \{s\}$, and
  - the parent of each node labeled with (s', j) is labeled with $(\mathit{predecessor}(s', i_j), j-1)$.
- The tree is infinite and finite-branching, and, hence, by König's Lemma, has an infinite branch $(s_{i_0}, i_0), (s_{i_1}, i_1), \ldots$, corresponding to an accepting run of $\mathcal{A}$:

$$\mathit{run}(s_{i_0}, i_0) \cdot \mathit{run}(s_{i_1}, i_1) \cdot \mathit{run}(s_{i_2}, i_2) \cdot \ldots$$

$\underline{\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{A}')}$:

- Let $\alpha \in \mathcal{L}(\mathcal{A})$.
- Let $r = s_0, s_1, \ldots$ be an accepting run of $\mathcal{A}$ on α.
- Let i be an index s.t. $s_i \in F$ and for all $j \geq i$ there exists a $k > j$, such that

$$\{s \in S \mid s_i \rightarrow^{\alpha(i,k)} s\} = \{s \in S \mid s_j \rightarrow^{\alpha(j,k)} s\}.$$

  This index exists:
  - "$\supseteq$" holds for all i, because there is a path through s_j.
  - Assume that for all i, there is a $j \geq i$ s.t for all $k > j$ "$\supsetneq$" holds. Then there exists an i' s.t. $\{s \in S \mid s_{i'} \rightarrow^{\alpha(i',k)} s\} = \emptyset$ for all $k > i'$. Contradiction.
- We define a run r' of $\mathcal{A}'$:

$$r' = P_0, \ldots, P_{i-1}, (\{s_i\}, \emptyset), (L_1, L_1'), (L_2, L_2') \ldots$$

  where $P_j = \{s \in S \mid p_0 \in I, p_0 \rightarrow^{\alpha(0,j)} s\}$, and L_j, L_j' are determinied by the definition of $\mathcal{A}'$.
- We show that r' is accepting. Assume otherwise, and let m be an index such that $L_n \neq L_n'$ for all $n \geq m$.
- Then let $j > m$ be some index with $s_j \in F$; hence $s_j \in L_j'$. There exists a $k > j$ such that $L_{k+1}' = \{s \in S \mid s_j \rightarrow^{\alpha(j,k)} s\} = \{s \in S \mid s_i \rightarrow^{\alpha(i,k)} s\} = L_{k+1}$.
- Contradiction.

■

Lemma 2 *For every semi-deterministic Büchi automaton $\mathcal{A}$ there exists a deterministic Muller automaton $\mathcal{A}'$ with $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$.*

Proof:

Let $\mathcal{A} = (N \uplus D, I, T, F)$, $d = |D|$, and let D be ordered by $<$. We construct the DMA $(S', \{s_0'\}, T', \mathcal{F})$:

- $S' = 2^N \times \{0, \ldots, 2d\} \rightarrow D \cup \{\llcorner\!\lrcorner\}$

- $s'_0 = (\{N \cap I\}, (d_1, d_2, \ldots, d_n, \llcorner\!\lrcorner, \ldots, \llcorner\!\lrcorner))$,
  where $d_i < d_{i+1}, \{d_1, \ldots, d_n\} = D \cap I\}$.
- $T' = \{((N_1, f_1), \sigma, (N_2, f_2)) \mid N_2 = pr_3(T \cap N_1 \times \{\sigma\} \times N)$
  $D' = pr_3(T \cap N_1 \times \{\sigma\} \times D)$
  $g_1 : n \mapsto d_2 \in D \Leftrightarrow f_1 : n \mapsto d_1 \in D \wedge d_1 \rightarrow^\sigma d_2$
  g_2: insort the elements of D' in the empty slots of g_1 (using $<$)
  f_2: delete every recurrance (leaving an *empty* slot)
- $\mathcal{F} = \{F' \subseteq S' \mid \exists i \in 1, \ldots, 2d$ s.t.
  $f(i) \neq \llcorner\!\lrcorner$ for all $(N', f) \in F'$ and
  $f(i) \in F$ for some $(N', f) \in F'\}$.

(... to be continued.)

$\blacksquare$

Bernd Finkbeiner
Sven Schewe

Date: June 6, 2008

Automata, Games and Verification: Lecture 6

Lemma 1 *For every semi-deterministic Büchi automaton $\mathcal{A}$ there exists a deterministic Muller automaton $\mathcal{A}'$ with $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$.*

Proof:

Let $\mathcal{A} = (N \uplus D, I, T, F)$, $d = |D|$, and let D be ordered by $<$. We construct the DMA $(S', \{s'_0\}, T', \mathcal{F})$:

- $S' = 2^N \times \{0, \ldots, 2d\} \to D \cup \{␣\}$
- $s'_0 = (\{N \cap I\}, (d_1, d_2, \ldots, d_n, ␣, \ldots, ␣))$,
  where $d_i < d_{i+1}, \{d_1, \ldots, d_n\} = D \cap I\}$.
- $T' = \{((N_1, f_1), \sigma, (N_2, f_2)) \mid N_2 = pr_3(T \cap N_1 \times \{\sigma\} \times N)$
  $D' = pr_3(T \cap N_1 \times \{\sigma\} \times D)$
  $g_1 : n \mapsto d_2 \in D \Leftrightarrow f_1 : n \mapsto d_1 \in D \wedge d_1 \to^\sigma d_2$
  g_2: insort the elements of D' in the empty slots of g_1 (using $<$),
  f_2: delete every recurrance (leaving an *empty* slot) $\}$;
- $\mathcal{F} = \{F' \subseteq S' \mid \exists i \in 1, \ldots, 2d$ s.t.
  $f(i) \neq ␣$ for all $(N', f) \in F'$ and
  $f(i) \in F$ for some $(N', f) \in F'\}$.

$\underline{\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{A}')}$:
If $\alpha \in \mathcal{L}(\mathcal{A})$, $\mathcal{A}$ has an accepting run $r = n_0 \ldots n_{j-1} d_j d_{j+1} d_{j+2} \ldots$
where $n_k \in N$ for $k < j$ and $d_k \in D$ for $k \geq j$.
Consider the run $r' = (N_0, f_0), (N_1, f_1), \ldots$ of $\mathcal{A}'$ on α.

- $n_k \in N_k$ for all $k < j$,
- for all $k \geq j$, $d_k = f_k(i)$ for some $i \leq 2d$,
- these i's are non-increasing, and hence stabilize eventually.
- for this stable i,
  $f(i) \neq ␣$ for all $(N', f) \in In(r')$ and $f(i) \in F$ for some $(N', f) \in In(r')$.
- $In(r') \in \mathcal{F}$.

$\underline{\mathcal{L}(\mathcal{A}') \subseteq \mathcal{L}(\mathcal{A})}$:
For $\alpha \in \mathcal{L}(\mathcal{A}')$, $\mathcal{A}'$ has an accepting run $r' = (N_0, f_0), (N_1, f_1), \ldots$.

- We pick an i and an accepting set $F' \in \mathcal{F}$ s.t.
  $f(i) \neq ␣$ for all $(N', f) \in F'$ and $f(i) \in F$ for some $(N', f) \in F'$.
- We pick a $j \in \omega$ such that $f_n(i) \neq ␣$ for all $n > j$.
- There is a run $r = s_0 s_1 \ldots s_j f_{j+1}(i) f_{j+2}(i) f_{j+3}(i) \ldots$ of $\mathcal{A}$ for α.
- r is accepting.

■

8 Linear-Time Temporal Logic (LTL)

1977: Amir Pnueli, *The temporal logic of programs* (Turing award 1996)

Syntax:

- Given a set of atomic propositions AP.
- Any atomic proposition $p \in AP$ is an LTL formula
- If φ, ψ are LTL formulars then so are
  - $\neg\varphi,\ \varphi \wedge \phi$,
  - $\bigcirc\varphi,\ \varphi\,\mathcal{U}\,\psi$

Abbreviations:
$\Diamond\varphi \ \equiv\ \mathit{true}\,\mathcal{U}\,\varphi$;
$\Box\,\varphi \ \equiv\ \neg(\Diamond\neg\varphi)$;
$\varphi\,\mathcal{W}\,\psi \ \equiv\ (\varphi\,\mathcal{U}\,\psi) \vee \Box\varphi$;

The *temporal operators:*

$\bigcirc$	X	Next
$\Box$	G	Always
$\Diamond$	F	Eventually
$\mathcal{U}$		Until
$\mathcal{W}$		Weak Until

Semantics: LTL formulas are interpreted over ω-words over 2^{AP}.
Notation: $\alpha, i \models \varphi$, where $\alpha \in (2^{AP})^\omega, i \in \omega$.

- $\alpha, i \models p$ if $p \in \alpha(i)$;
- $\alpha, i \models \neg\varphi$ if $\alpha, i \not\models \varphi$;
- $\alpha, i \models \varphi \wedge \psi$ if $\alpha, i \models \varphi$ and $\alpha, i \models \psi$;
- $\alpha, i \models \bigcirc\varphi$ if $\alpha, i+1 \models \varphi$
  $\alpha, i \models \varphi\,\mathcal{U}\,\psi$ if there is some $j \geq i$ s.t. $\alpha, j \models \psi$ and for all $i \leq k < j$: $\alpha, k \models \varphi$

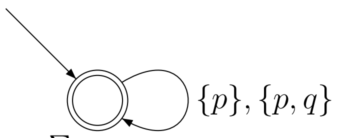
Abbreviation: $\alpha \models \varphi \ \equiv\ \alpha, 0 \models \varphi$

Definition 1

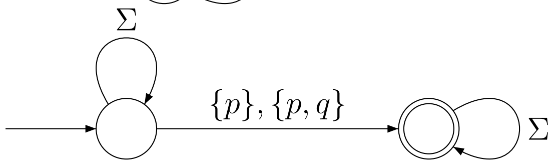
- *models*(φ)=$\{\alpha \in (2^{AP})^w \mid \alpha \models \varphi\}$
- *an LTL formula* φ *is* satisfiable *if models*(φ) $\neq \emptyset$
- *an LTL formula* φ *is* valid *if models*(φ) $= (2^{AP})^\omega$

Example: LTL formulas with $AP = \{p, q\}$:

- Safety: $\Box p$



- Guarantee: $\Diamond p$



✚

There are Büchi recognizable languages that are not LTL-definable.
Example: $(\emptyset\emptyset)^*\{p\}^\omega$

Definition 2 *A language* $L \subseteq \Sigma^\omega$ *is* non-counting *iff*
$\exists n_0 \in \omega \; . \; \forall n \geq n_0 \; . \; \forall u, v \in \Sigma^*, \gamma \in \Sigma^\omega \; .$
$uv^n\gamma \in L \;\Leftrightarrow\; uv^{n+1}\gamma \in L$

Example: $L = (\emptyset\emptyset)^*\{p\}^\omega$ is counting. For every $\emptyset^n\{p\}^\omega \in L$, $\emptyset^{n+1}\{p\}^\omega \notin L$. ✚

Theorem 1 *For every LTL-formula φ, models(φ) is non-counting.*

Proof:

Structural induction on φ:

- $\varphi = p$: choose $n_0 = 1$.
- $\varphi = \varphi_1 \wedge \varphi_2$: By IH, φ_1 defines non-counting language with threshold $n_0' \in \omega$, φ_2 with n_0''; choose $n_0 = \max(n_0', n_0'')$;
- $\varphi = \neg\varphi_1$: choose $n_0 = n_0'$.
- $\varphi = \bigcirc\varphi_1$: choose $n_0 = n_0' + 1$.
  - We show for $n \geq n_0$: $uv^n\gamma \models \bigcirc\varphi \;\Leftrightarrow\; uv^{n+1}\gamma \models \bigcirc\varphi$.
  - Case $u \neq \epsilon$, i.e., $u = au'$ for some $a \in \Sigma, u' \in \Sigma^*$:

    $au'v^n\gamma \models \bigcirc\varphi$
    iff $u'v^n\gamma \models \varphi$
    iff $u'v^{n+1}\gamma \models \varphi$ (IH)
    iff $au'v^{n+1}\gamma \models \bigcirc\varphi$.
  - Case $u = \epsilon, v = av'$ for some $a \in \Sigma, v' \in \Sigma^*$:

    $(av')^n\gamma \models \bigcirc\varphi$
    iff $(av')(av')^{n-1}\gamma \models \bigcirc\varphi$
    iff $v'(av')^{n-1}\gamma \models \varphi$
    iff $v'(av')^n\gamma \models \varphi$ (IH)
    iff $(av')^{n+1}\gamma \models \bigcirc\varphi$.

- $\varphi = \varphi_1 \,\mathcal{U}\, \varphi_2$: choose $n_0 = \max(n_0', n_0'') + 1$.
  Claim: for $n \geq n_0$: $uv^n\gamma \models \varphi_1 \,\mathcal{U}\, \varphi_2 \;\Rightarrow\; uv^{n+1}\gamma \models \varphi_1 \,\mathcal{U}\, \varphi_2$.
  - $uv^n\gamma \models \varphi_1 \,\mathcal{U}\, \varphi_2 \;\Rightarrow\; \exists j \,.\, uv^n\gamma, j \models \varphi_2$ and $\forall i < j \,.\, uv^n\gamma, i \models \varphi_1$.
  - Case $j \leq |u|$:
    by IH, $uv^{n+1}, j \models \varphi_2$ and for all $i < j \,.\, uv^{n+1}, i \models \varphi_1$;
  - Case $j > |u|$:
    $uv^{n+1}\gamma, j + |v| \models \varphi_2$;
    for all $|u| + |v| \leq i < j + |v| \,.\, uv^{n+1}\gamma, i \models \varphi_1$;
    By (IH), for all $i < |u| + |v| \,.\, uvv^n\gamma, i \models \varphi_1$, because $uvv^{n-1}\gamma, i \models \varphi_1$.

  Claim: for $n \geq n_0$: $uv^{n+1}\gamma \models \varphi_1 \,\mathcal{U}\, \varphi_2 \;\Rightarrow\; uv^n\gamma \models \varphi_1 \,\mathcal{U}\, \varphi_2$
  - $uv^{n+1}\gamma \models \varphi_1 \,\mathcal{U}\, \varphi_2 \;\Rightarrow\; \exists j \,.\, uv^{n+1}\gamma, j \models \varphi_2$ and $\forall i < j \,.\, uv^{n+1}\gamma, i \models \varphi_1$.
  - Case $j \leq |u| + |v|$:
    by IH, $uvv^{n-1}, j \models \varphi_2$ and for all $i < j \,.\, uvv^{n-1}, i \models \varphi_1$;
  - Case $j > |u| + |v|$:
    $uv^n\gamma, j - |v| \models \varphi_2$;
    for all $|u| + |v| \leq i < j \,.\, uv^n\gamma, i \models \varphi_1$;
    By (IH), for all $i < |u| + |v| \,.\, uvv^{n-1}\gamma, i \models \varphi_1$, because $uvv^n\gamma, i \models \varphi_1$.

■

Bernd Finkbeiner
Sven Schewe

Date: June 12, 2008

Automata, Games and Veri cation: Lecture 7

9 Quantified Propositional Temporal Logic (QPTL)

Syntax: LTL formula $\mid \varphi \wedge \varphi \mid \neg\varphi \mid \exists p.\ \varphi$

Semantics:

$$, i \models \exists q.\varphi \quad \text{i} \quad \text{there is an } \ ' \text{ with}$$
$$'(j) \cap (AP \setminus \{q\}) = \ (j) \cap (AP \setminus \{q\}) \text{ for all } j \in \omega,$$
$$\text{s.t. } \ ', i \models \varphi.$$

Example: $L = (\emptyset\emptyset) \ \{p\}^\omega$ is QPTL-de nable:

$\exists q.\ (q \wedge \Box(q \leftrightarrow \neg q) \wedge \Box(p \rightarrow \bigcirc p) \wedge \Box(\bigcirc p \leftrightarrow p \vee q))$ ✚

Theorem 1 *For every Buchi automaton $\mathcal{A}$ over $\ = 2^{AP}$ there exists a QPTL formula φ such that models$(\varphi) = \mathcal{L}(\mathcal{A})$.*

Proof:

Let $S = \{s_1, s_2, \dots, s_n\}$ and $AP' = AP \cup \{at_{s_1}, \dots, at_{s_n}\}$.

$$\begin{aligned}
\varphi \ := \ \exists at_{s_1}, \dots, at_{s_n} \quad . \quad & \bigvee_{s \in I} at_s \\
\wedge \quad & \Box \left(\bigvee_{(s_i, A, s_j) \in T} at_{s_i} \wedge \bigcirc at_{s_j} \wedge \left(\bigwedge_{p \in A} p \right) \wedge \left(\bigwedge_{p \in AP \setminus A} \neg p \right) \right) \\
\wedge \quad & \Box \left(\bigvee_{i=1}^{n} \bigwedge_{j \neq i} \neg(at_{s_i} \wedge at_{s_j}) \right) \\
\wedge \quad & \Box\Diamond \bigvee_{s_i \in F} at_{s_i}
\end{aligned}$$

■

10 Monadic Second-Order Theory of One Successor (S1S)

Syntax:

rst-order variable set $V_1 = \{x, y, \dots\}$

second-order variable set $V_2 = \{X, Y, \ldots\}$

Terms t:

$$t ::= 0 \mid x \mid S(t)$$

Formulas φ:

$$\varphi ::= t \in X \mid t_1 = t_2 \mid \neg\varphi \mid \varphi_0 \vee \varphi_1 \mid \exists x.\varphi \mid \exists X.\varphi$$

Abbreviations:

$\forall X.\ \varphi \ := \ \neg\exists X.\ \neg\varphi;$

$x \notin Y \ := \ \neg(x \in Y);$

$x \neq y \ := \ \neg(x = y).$

Semantics:

rst-order valuation $\ _1 : V_1 \rightarrow \omega$

second-order valuation $\ _2 : V_2 \rightarrow 2^\omega$

Semantics of terms:

$[0]\ _1 = 0$

$[x]\ _1 = \ _1(x)$

$[S(t)\ _1] = [t]\ _1 + 1$

Semantics of formulas:

$_1,\ _2 \models t \in X$ i $[t]\ _1 \in \ _2(X)$

$_1,\ _2 \models t_1 = t_2$ i $[t_1]\ _1 = [t_2]\ _1$

$_1,\ _2 \models \neg$ i $_1,\ _2 \not\models$

$_1,\ _2 \models \ _0 \vee \ _1$ i $_1,\ _2 \models \ _0$ or $_1,\ _2 \models \ _1$

$_1,\ _2 \models \exists x.\ \varphi$ i there is an $a \in \omega$ s.t.

$${}'_1(y) = \begin{cases} \ _1(y) \text{ if } y \neq x \\ a \text{ otherwise} \end{cases}$$

and ${}'_1,\ _2 \models \varphi$.

$_1,\ _2 \models \exists X.\ \varphi$ i there is an $A \quad \omega$ s.t.

$${}'_2(Y) = \begin{cases} \ _2(Y) \text{ if } Y \neq X \\ A \text{ otherwise} \end{cases}$$

and $_1,\ {}'_2 \models \varphi$

Example:

$$\begin{aligned}
X \quad Y: &\quad \forall z.\ (z \in X \rightarrow z \in Y);\\
X = Y: &\quad X \quad Y \wedge Y \quad X;\\
Su \quad (X): &\quad \forall y.\ (y \in X \rightarrow S(y) \in X);\\
x \quad y: &\quad \forall Z.\ (x \in Z \wedge Su \quad (Z)) \rightarrow y \in Z;\\
Fin(X): &\quad \exists Y.\ ((X \quad Y \wedge \exists z \quad z \notin Y \wedge \forall z.\ (z \notin Y \rightarrow S(z)) \notin Y);
\end{aligned}$$

✚

De nition 1 *For a S1S formula φ, models(φ) $= \{ \quad _1, \, _2 \in (2^{V_1 \cup V_2})^\omega \mid \quad _1, \, _2 \models \varphi\}$, where $x \in \quad (j)$ i $j = \, _1(x)$, and $X \in \quad (j)$ i $j \in \, _2(X)$.*

De nition 2 *A language L is LTL/QPTL/S1S-definable if there is a LTL/QPTL/S1S formula φ with models(φ) $= L$.*

Theorem 2 *Every QPTL-definable language is S1S-definable.*

Proof:

For every QPTL-formula φ over AP and every S1S-term t over $V_1 = \emptyset$, we de ne a S1S formula $T(\varphi, t)$ over $V_1 = \emptyset$, $V_2 = AP$, such that, for all $\quad \in (2^{AP})^\omega$,

$$, [t]_{\ _1} \models_{\text{QPTL}} \varphi \quad \text{i} \quad \ _1, \ _2 \models_{\text{S1S}} T(\varphi, t),$$

where $\ _2 : P \mapsto \{i \in \omega \mid P \in \quad (i)\}$.

$T(P, t) = t \in P$, for $P \in AP$;
$T(\neg\varphi, t) = \neg T(\varphi, t)$;
$T(\varphi \vee \quad , t) = T(\varphi, t) \vee T(\quad , t)$
$T(\bigcirc\varphi, t) = T(\varphi, S(t))$
$T(\varphi\, \mathcal{U} \quad , t) = \exists y.(y \quad t \wedge T(\quad , y) \wedge \neg\exists z.(x \quad z < y \wedge T(\neg\varphi, z)))$
$T(\exists P\ \varphi, t) = \exists P.\ T(\varphi, t)$.

models(φ) = models($T(\varphi, 0)$). ■

Theorem 3 *Every S1S-definable language is Buchi-recognizable.*

Proof:

Let φ be a S1S-formula.

1. Rewrite φ into normal form
   $\varphi ::= \ 0 \in X \mid x \in Y \mid x = 0 \mid x = y \mid x = S(y) \mid$
   $\quad \neg\varphi \mid \varphi \vee \quad \mid \exists x.\ \varphi \mid \exists X.\ \varphi.$
   using the following rewrite rules:

$$\begin{aligned}
S(t) \in X &\quad \mapsto \quad \exists y.\ y = S(t) \wedge y \in X\\
S(t) = S(t') &\quad \mapsto \quad t = t'\\
S(t) = x &\quad \mapsto \quad x = S(t)\\
t = S(S(t')) &\quad \mapsto \quad \exists y.\ y = S(t') \wedge t = S(y)
\end{aligned}$$

2. Rename bound variables to obtain unique variables.

   Example:

$$\exists x.(S(S(y)) = x \land \exists x\ (S(x) \in X_0))$$

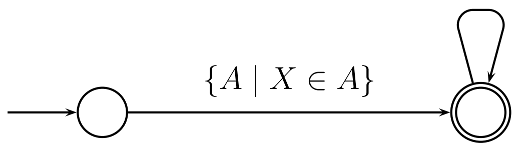
   is rewritten to

$$\exists x_0.\ \exists x_1.x_0 = S(x_1) \land x_1 = S(y) \land \exists x_2 \exists x_3.x_3 = S(x_2) \land x_3 \in X_0$$

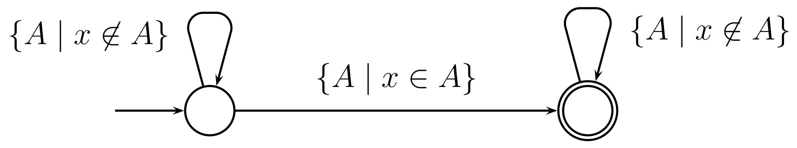
3. Construct Buchi automaton:

   Base cases:

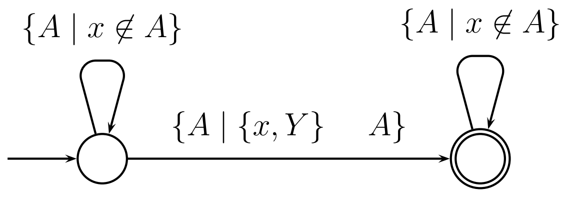
   $0 \in X$:



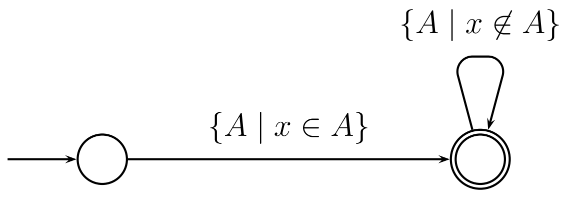
   For every $x \in V_1$, intersect with $\mathcal{A}_x$:



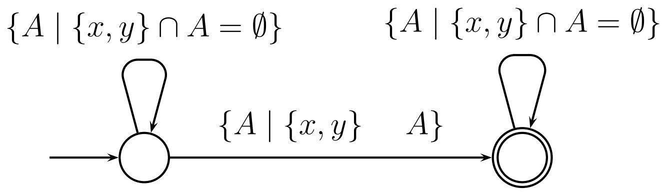
   $x \in Y$:



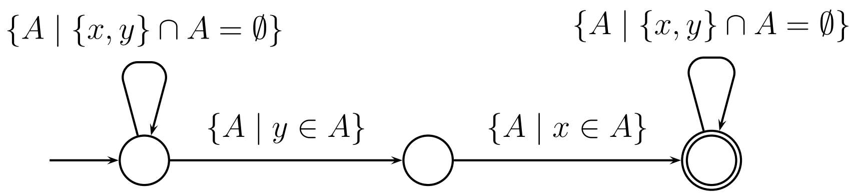
   $x = 0$:



$x = y$:


$\{A \mid \{x, y\} \cap A = \emptyset\}$
$\{A \mid \{x, y\} \cap A = \emptyset\}$
$\{A \mid \{x, y\} \quad A\}$

$x = S(y)$:


$\{A \mid \{x, y\} \cap A = \emptyset\}$
$\{A \mid \{x, y\} \cap A = \emptyset\}$
$\{A \mid y \in A\}$
$\{A \mid x \in A\}$

Inductive step:

$\varphi \vee \quad$: language union,
$\neg\varphi$: complement (and intersection with all $\mathcal{A}_x$),
$\exists x.\ \varphi$: projection (and intersection with $\mathcal{A}_x$),
$\exists X.\ \varphi$: projection.

■

Bernd Finkbeiner
Sven Schewe

Date: June 19, 2008

Automata, Games and Verification: Lecture 8

11 Weak Monadic Second-Order Theory of One Successor (WS1S)

Syntax: same as S1S;

Semantics: same as S1S; except:
$\sigma_1, \sigma_2 \models \exists X.\varphi$ iff there is a **finite** $A \subseteq \omega$ s.t.

$$\sigma_2'(X) = \begin{cases} \sigma_2(X) \text{ if } X \neq X_i \\ A \text{ otherwise} \end{cases}$$

and $\sigma_1, \sigma_2' \models \varphi$.

Theorem 1 *A language is WS1S-definable iff it is S1S-definable.*

Proof:

($\Rightarrow$): Quantifier relativization:

$$\begin{array}{lcl} \forall X \ldots & \mapsto & \forall X.\ \mathrm{Fin}(X) \rightarrow \ldots \\ \exists X \ldots & \mapsto & \forall X.\ \mathrm{Fin}(X) \wedge \ldots \end{array}$$

($\Leftarrow$):

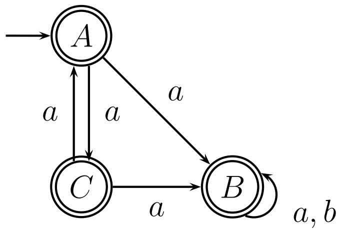
- Let φ be an S1S-formula.
- Let $\mathcal{A}$ be a Büchi automaton with $\mathcal{L}(\mathcal{A}) = \mathrm{models}(\varphi)$.
- Let $\mathcal{A}'$ be a deterministic Muller automaton with $\mathcal{L}(\mathcal{A}') = \mathcal{L}(\mathcal{A})$..
- By the characterization of deterministic Muller languages, $\mathcal{L}(\mathcal{A}')$ is a boolean combination of languages $\overrightarrow{W}$, where W is finite-word recognizable.
- Let $\psi(y)$ be a WS1S formula that defines the words whose prefix up to position y is in W.
- $\varphi' := \forall x.\ \exists y.\ (x < y \wedge \psi(y))$.

■

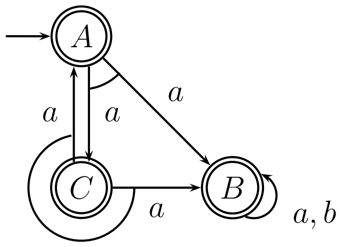
12 Alternating Automata

Example:

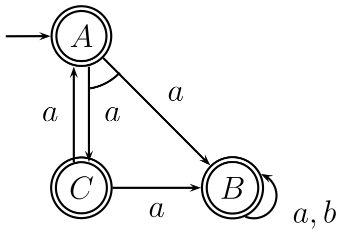
- Nondeterministic automaton, $L = a(a+b)^\omega$, existential branching mode:



- $\forall$-automaton, $L = a^\omega$, universal branching mode:



- Alternating automaton, both branching modes (arc between edges indicates universal branching mode), $L = aa(a+b)^\omega$



✚

Definition 1 *The* positive Boolean formulas *over a set X, denoted $\mathbb{B}^+(X)$, are the formulas built from elements of X, conjunction $\wedge$, disjunction $\vee$, true and false.*

Definition 2 *A set $Y \subseteq X$* satisfies *a formula $\varphi \in B^+(X)$, denoted $Y \models \varphi$, iff the truth assignment that assigns true to the members of Y and false to the members of $X \setminus Y$ satisfies φ.*

Definition 3 *An* alternating Büchi automaton *is a tuple $\mathcal{A} = (S, s_0, \delta, F)$, where:*

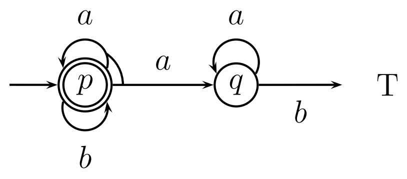
- *S is a finite set of states,*
- *$s_0 \in S$ is the initial state,*
- *$F \subseteq S$ is the set of accepting states, and*
- *$\delta : S \times \Sigma \to \mathbb{B}^+(S)$ is the transition function.*

A tree T over a set of *directions* D is a prefix-closed subset of D^*. The empty sequence ϵ is called the *root*. The children of a node $n \in T$ are the nodes children$(n) = \{n \cdot d \in T \mid d \in D\}$. A Σ-labeled tree is a pair (T, l), where $l : T \to \Sigma$ is the labeling function.

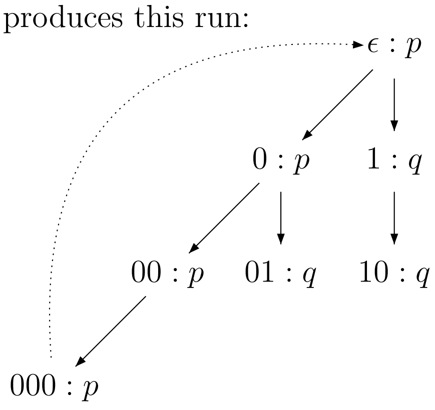
Definition 4 *A* run *of an alternating automaton on a word $\alpha \in \Sigma^\omega$ is an S-labeled* tree *$\langle T, r\rangle$ with the following properties:*

- *$r(\epsilon) = s_0$ and*
- *for all $n \in T$, if $r(n) = s$, then $\{r(n') \mid n' \in children(n)\}$ satisfies $\delta(s, \alpha(|n|))$.*

Example: $L = (\{a, b\}^* \, b)^\omega$



$S = \{p, q\}$
$F = \{p\}$
$\delta(p, a) = p \wedge q$
$\delta(p, b) = p$
$\delta(q, a) = q$
$\delta(q, b) = \mathrm{T}$
example word $w = (aab)^\omega$ produces this run:



(the dotted line means that the same tree would repeat there) ✚

Definition 5 *A* branch *of a tree T is a maximal sequence of words $n_0, n_1, n_2, \cdots$ such that $n_0 = \epsilon$ and n_{i+1} is a child of n_i for $i \geq 0$.*

Notation: Infinity set of a branch β in a run tree (T, r):

$$In(\beta) = \{s \in S \mid \forall i \exists j : j \geq i \wedge r(\beta(j)) = s\}$$

Definition 6 *A run (T, r) is* accepting *iff, for every infinite branch $Y' \subseteq Y$,*

$$In(Y') \cap F \neq \emptyset.$$

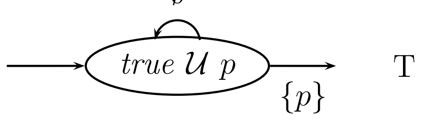
Theorem 2 *For every LTL formula φ, there is an alternating Büchi automaton $\mathcal{A}$ with $\mathcal{L}(\mathcal{A}) = models(\varphi)$*

Proof:

- $S = \text{closure}(\varphi) := \{\psi, \neg\psi \mid \psi \text{ is subformula of } \varphi\}$;
- $s_0 = \varphi$;
- $\delta(p,a) = \mathit{true}$ if $p \in a$, false if $p \notin a$;
  $\delta(\neg p,a) = \mathit{false}$ if $p \in a$, true if $p \notin a$;
  $\delta(\mathit{true},a) = \mathit{true}$;
  $\delta(\mathit{false},a) = \mathit{false}$;
- $\delta(\psi_1 \wedge \psi_2, a) = \delta(\psi_1,a) \wedge \delta(\psi_2,a)$;
- $\delta(\psi_1 \vee \psi_2, a) = \delta(\psi_1,a) \vee \delta(\psi_2,a)$;
- $\delta(\bigcirc \psi, a) = \psi$;
- $\delta(\psi_1 \;\mathcal{U}\; \psi_2, a) = \delta(\psi_1,a) \vee (\delta(\psi_2,a) \wedge \psi_1 \;\mathcal{U}\; \psi_2)$;
- $\delta(\neg\psi, a) = \overline{\delta(\psi,a)}$;
- $\overline{\psi} = \neg\psi$ for $\psi \in S$;
- $\overline{\neg\psi} = \psi$ for $\psi \in S$;
- $\overline{\psi_1 \wedge \psi_2} = \overline{\alpha} \vee \overline{\beta}$;
- $\overline{\psi_1 \vee \psi_2} = \overline{\alpha} \wedge \overline{\beta}$;
- $\overline{\mathit{true}} = \mathit{false}$;
- $\overline{\mathit{false}} = \mathit{true}$;
- $F = \{\neg(\psi_1 \;\mathcal{U}\; \psi_2) \in \text{closure}(\varphi)\}$

For a subformula ψ of φ let $\mathcal{A}_\varphi^\psi$ be the automaton A_φ with initial state ψ.
Claim: $\alpha \in \mathcal{L}(\mathcal{A}_\varphi^\psi) \;\Leftrightarrow\; \alpha \in \mathit{models}(\psi)$. Proof by structural induction. ■

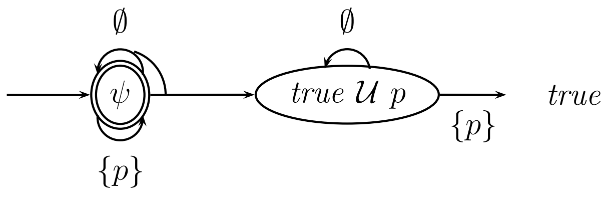
Example: $\varphi := \Diamond p \equiv (\mathit{true} \;\mathcal{U}\; p)$
$S = \{\mathit{true} \;\mathcal{U}\; p, \neg(\mathit{true} \;\mathcal{U}\; p), \mathit{true}, \neg\mathit{true}, p, \neg p\}$
$\delta(\mathit{true} \;\mathcal{U}\; p, \emptyset) = \delta(p,\emptyset) \vee (\delta(\mathit{true},\emptyset) \wedge \mathit{true} \;\mathcal{U}\; p) = \mathit{true} \;\mathcal{U}\; p$
$\delta(\mathit{true} \;\mathcal{U}\; p, \{p\}) = \delta(p,\{p\}) \vee (\delta(\mathit{true},\{p\}) \wedge \mathit{true} \;\mathcal{U}\; p) = \mathrm{T}$



$\varphi := \Box\Diamond p \equiv \neg(\mathit{true} \;\mathcal{U}\; \neg(\mathit{true} \;\mathcal{U}\; p))$

$$\begin{aligned}
\delta(\varphi,a) &= \overline{\delta(\neg(\mathit{true} \;\mathcal{U}\; p),a) \vee (\delta(\mathit{true},a) \wedge \mathit{true} \;\mathcal{U}\; \neg(\mathit{true} \;\mathcal{U}\; p))} \\
&= \delta(\mathit{true} \;\mathcal{U}\; p,a) \wedge \neg(\mathit{true} \;\mathcal{U}\; \neg(\mathit{true} \;\mathcal{U}\; p)) \\
&= (\delta(p,a) \vee (\delta(\mathit{true},a) \wedge \mathit{true} \;\mathcal{U}\; p)) \wedge \varphi \\
&= (\delta(p,a) \vee \mathit{true} \;\mathcal{U}\; p) \wedge \varphi
\end{aligned}$$

$\delta(\varphi,\emptyset) = \mathit{true} \;\mathcal{U}\; p \wedge \varphi$
$\delta(\varphi,\{p\}) = \varphi$



$\emptyset$
ψ
$\{p\}$
$\emptyset$
$true\ \mathcal{U}\ p$
$\{p\}$
$true$

✚

Bernd Finkbeiner
Sven Schewe

Date: June 26, 2008

Automata, Games and Verification: Lecture 9

Definition 1 *Two nodes $x_1, x_2 \in T$ in a run tree (T, r) are* similar *if $|x_1| = |x_2|$ and $r(x_1) = r(x_2)$.*

Definition 2 *A run tree (T, r) is* memoryless *if for all similar nodes x_1 and x_2 and for all $y \in D^*$ we have that ($x_1 \cdot y \in T$ iff $x_2 \cdot y \in T$) and $r(x_1 \cdot y) = r(x_2 \cdot y)$.*

Theorem 1 *If an alternating Büchi Automaton $\mathcal{A}$ accepts a word α, then there exists a memoryless accepting run of $\mathcal{A}$ on α.*

Proof:

- Let (T, r) be an accepting run tree on α with directions D.
- We define $\gamma : T \to \omega$ (measures the number of steps since the last visit to F):
  - $\gamma(\epsilon) = 0$
  - $\gamma(x \cdot d) = \begin{cases} \gamma(x) + 1 & \text{if } x \notin F; \\ 0 & \text{otherwise;} \end{cases}$
- We define $\Delta : S \times \omega \to T$:
  $\Delta(s, n) =$ leftmost $y \in T$ with $|y| = n, r(y) = s$ and $(\forall z \in T, |z| = n \wedge r(z) = s \Rightarrow \gamma(z) \leq \gamma(y))$.
- We define (T', r'):
  - $\epsilon \in T$, $r'(\epsilon) = r(\epsilon)$;
  - for $n \in T'$, $d \in D$,
    $x \cdot d \in T'$ iff $\Delta(r'(n), |n|) \cdot d \in T$;
    $r'(n \cdot d) = r(\Delta(r'(n), |n|) \cdot d)$

Claim 1: (T', r') is a run of $\mathcal{A}$ on α.

- $r'(\epsilon) = r(\epsilon) = s_0$
- For $n \in T'$, let $q_n = \Delta(r'(n), |n|)$.
- For every $n \in T'$, $\{r(q_n \cdot d) \mid d \in D, q_n \cdot d \in T\} \models \delta(r(q_n), \alpha(|q_n|))$
  and therefore $\{r'(n \cdot d) \mid d \in D, n \cdot d \in T'\} \models \delta(r'(n), \alpha(|n|))$.

Claim 2: If (T, r) is accepting, then so is (T', r'). Proof by contradiction:

- Suppose (T', r') is not accepting, then there is an infinite branch $\pi : n_0, n_1, n_2, \ldots \in T'$ and $\exists k \in \omega$ such that $\forall j \geq k : r'(b_j) \notin F$.
- Let $m_i = \Delta(r'(n_i), |n_i|)$ for $i \geq k$.
- **Claim 2.1** : For every $m \in T'$, $\gamma(m) \leq \gamma(\Delta(r'(m), |m|))$. Proof by induction on the length of m:

- for $m = \epsilon$, $\gamma(m) = 0$
- for $m = m' \cdot d$ (where $d \in D$),
  - if $r(m') \in F$, then $\gamma(m) = 0$
  - if $r(m') \notin F$, then

$$\begin{array}{ll} & \gamma(\Delta(r'(m' \cdot d), |m' \cdot d|)) \\ \geq & \quad (\Delta \text{ definition}) \\ & \gamma(\Delta(r'(m'), |m'|) \cdot d) \\ = & \quad (\gamma \text{ definition}) \\ & 1 + \gamma(\Delta(r'(m'), |m'|)) \\ \geq & \quad (\text{induction hypothesis}) \\ & 1 + \gamma(m') \\ = & \quad (\gamma \text{ definition }) \\ & \gamma(m' \cdot d) \end{array}$$

- We have,

$$\begin{array}{ccccc} \gamma(n_k) & < & \gamma(n_{k+1}) & < & \dots \\ /\!\wedge & & /\!\wedge & & \\ \gamma(m_k) & < & \gamma(m_{k+1}) & < & \dots \end{array}$$

  So, for any $k' > k, \gamma(m_k) \geq k' - k$.

  Since T is finitely branching, there must be a branch with an infinite suffix of non-F labeled positions. This contradicts our assumption that (T, r) is accepting.

■

Definition 3 *A* run DAG *of an alternating Büchi Automaton $\mathcal{A}$ on word α is a DAG (V, E), where*

- $V \subseteq S \times \omega$
- $E \subseteq \bigcup_{i \in \omega} (S \times \{i\}) \times (S \times \{i + 1\})$;
- $(s_0, 0) \in V$
- $\forall (s, i) \in V \ . \ \exists Y \subseteq S$ *s.t.*
  $Y \models \delta(s, \alpha(i)), Y \times \{i + 1\} \subseteq V$ *and* $\{(s, i)\} \times (Y \times \{i + 1\}) \subseteq E$.

Example:

$$\begin{tikzcd}[column sep=small]
 & & & & p \arrow[dl] \arrow[dr] & & & & \\
 & & & p \arrow[dl] \arrow[dr] & & q \arrow[dr] & & & \\
 & & p \arrow[dl] \arrow[dr] & & q \arrow[dr] & & q \arrow[dr] & & \\
 & p \arrow[dl] \arrow[dr] & & q \arrow[dr] & & q \arrow[dr] & & q \arrow[dr] & \\
\maltese & & {} & & {} & & {} & & {}
\end{tikzcd}
\qquad
\begin{tikzcd}
(p,0) \arrow[d] \arrow[dr] & \\
(p,1) \arrow[d] \arrow[dr] & (q,1) \arrow[d] \\
(p,2) \arrow[d] \arrow[dr] & (q,2) \arrow[d] \\
{} & {}
\end{tikzcd}$$

Notation: Level $((V,E),i) = \{s \in S \mid (s,i) \in V\}$

Definition 4 *A run DAG is* accepting *if every path has infinitely many visits to $F \times \omega$.*

Corollary 1 *A word α is accepted by an alternating Büchi automaton $\mathcal{A}$ iff $\mathcal{A}$ has an accepting run DAG on α.*

Theorem 2 (Miyano and Hayashi, 1984) *For every alternating Büchi automaton $\mathcal{A}$, there exists a nondeterministic Büchi automaton $\mathcal{A}'$ with $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$.*

Proof:

- $S' = 2^S \times 2^S$;
- $I' = \{(\{s_0\}, \emptyset)\}$;
- $F' = \{(X, \emptyset) \mid X \subseteq S\}$;
- $T' = \{((X,\emptyset), \sigma, (X', X' - F)) \mid X' \models \bigwedge_{s \in X} \delta(s,\sigma)\}$
  $\cup \{((x,W), \sigma, (X', W' \setminus F)) \mid W \neq \emptyset, W' \subseteq X', X' \models \bigwedge_{s \in X} \delta(s,\sigma),$
  $W' \models \bigwedge_{s \in W} \delta(s,\sigma)\}$.

$\mathcal{L}(\mathcal{A}') \subseteq \mathcal{L}(\mathcal{A})$:

- Let $\alpha \in L(\mathcal{A}')$ with accepting run

  $$r' : (X_0, W_0)(X_1, W_1)(X_2, W_2) \ldots$$

  where $W_0 = \emptyset, X_0 = \{s_0\}$.
- We construct the run DAG (V,E) for $\mathcal{A}$ on α:
  - $V = \bigcup_{i \in \omega} X_i \times \{i\}$;
  - $E = \bigcup_{i \in \omega} (\bigcup_{x \in X_i \setminus W_i} \{(x,i)\} \times (X_{i+1} \times \{i+1\})$
    $\cup \bigcup_{x \in W_i} \{(x,i)\} \times \{(X_{i+1} \cap (F \cup W_{i+1})) \times \{i+1\})$.
- (V,E) is an accepting run DAG:
  - $(s_0, 0) \in V$;
  - for $(x,i) \in V$:
    - if $x \in X_i \setminus W_i$, $X_{i+1} \models \delta(x, \alpha(i))$;
    - if $x \in W_i$, $X_{i+1} \cap (F \cup W_{i+1}) \models \delta(x, \alpha(i))$.

  - Every path through the run DAG visits F infinitely often (otherwise $W_i = \emptyset$ only for finitely many i).

$\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{A}')$:

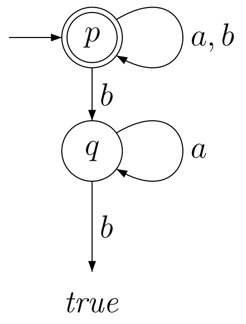
- Let $\alpha \in L(A')$ and (V, E) an accepting run DAG of $\mathcal{A}'$ on α.
- We construct a run

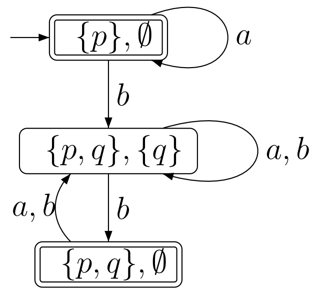
$$r' : (X_0, W_0)(X_1, W_1)(X_2, W_2) \dots$$

  on $\mathcal{A}$ as follows:
  - $X_0 = \{s_0\}, W_0 = \emptyset$;
  - for $i > 0$, $X_i = \mathit{Level}((V, E), i)$
    - if $W_i = \emptyset$ then $W_{i+1} = X_{i+1} \setminus F$,
    - otherwise, $W_{i+1} := \{y' \in S \setminus F \mid \exists (y, i) \in V, ((y, i), (y', i+1)) \in E, y \in W_i\}$.
- r' is an accepting run:
  - starts with $(\{s_0\}, \emptyset)$
  - obeys T':
    - for $x \in X_i \setminus W_i$, $X_{i+1} \models \delta(x, \alpha(i))$;
    - for $x \in W_i$, $X_{i+1} \cap (F \cup W_{i+1}) \models \delta(x, \alpha(i))$.
  - r' is accepting (otherwise there exists a path in (V, E) that is not accepting).

■

Example: We translate the following *universal* automaton (all branchings are conjunctions) into an equivalent nondeterministic automaton:





✚

Corollary 2 *A language is ω-regular iff it is recognizable by an alternating Büchi automaton.*

Proof:

Translation from nondeterministic Büchi automaton $(S, \{s_0\}, T, F)$ to alternating Büchi automaton (S, s_0, δ, F) with

- $\delta(s,\sigma) = \bigvee_{s' \in pr_3(T \cap \{s\} \times \{\sigma\} \times S)} s'$ for all $s \in S$

$\blacksquare$

Corollary 3 Satisfiability *of an LTL formula φ can be checked in time exponential in the length of φ.*

Corollary 4 Validity *of an LTL formula φ can be checked in time exponential in the length of φ.*

Comment: Acceptance of a word α by an alternating Büchi automaton can also be characterized by a game:

- Positions of player Blue: $B = S \times \omega$;
- Positions of player Green: $G = 2^S \times \omega$;
- Edges: $\{((s,i),(X,i)) \mid X \models \delta(s,\alpha(i))\}$
  $\cup \{((X,i),(s,i+1)) \mid s \in X\}$

Blue wins a play iff $F \times \omega$ is visited infinitely often.

The word α is accepted iff Blue has a strategy to win the game from position $(s_0, 0)$.
End Comment

Bernd Finkbeiner
Sven Schewe

Date: July 3, 2008

Automata, Games and Verification: Lecture 10

13 Games

Definition 1 *A* game arena *is a triple $\mathcal{A} = (V_0, V_1, E)$, where*

- *V_0 and V_1 are disjoint sets of positions, called the positions of player 0 and 1,*
- *$E \subseteq V \times V$ for set $V = V_0 \uplus V_1$ of game positions,*
- *every position $p \in V$ has at least one outgoing edge $(p, p') \in E$.*

Definition 2 *A* play *is an infinite sequence $\pi = p_0p_1p_2 \ldots \in V^\omega$ such that $\forall i \in \omega \,.\, (p_i, p_{i+1}) \in E$.*

Definition 3 *A* strategy *for player σ is a function $f_\sigma : V^* \cdot V_\sigma \to V$ s.t. $(p, p') \in E$ whenever $f(u \cdot p) = p'$.*

Definition 4 *A play $\pi = p_0, p_1, \ldots$* conforms to *strategy f_σ of player σ if $\forall i \in \omega \,.$ if $p_i \in V_\sigma$ then $p_{i+1} = f_\sigma(p_0, \ldots, p_i)$.*

Definition 5

- *A* reachability game *$\mathcal{G} = (\mathcal{A}, R)$ consists of a game arena and a winning set of positions $R \subseteq V$. Player 0 wins a play $\pi = p_0p_1 \ldots$ if $p_i \in R$ for some $i \in \omega$, otherwise Player 1 wins.*
- *A* Büchi game *$\mathcal{G} = (\mathcal{A}, F)$ consists of an arena $\mathcal{A}$ and a set $F \subseteq V$. Player 0 wins a play π if $In(\pi) \cap F \neq \emptyset$, otherwise Player 1 wins.*
- *A* Parity game *$\mathcal{G} = (\mathcal{A}, c)$ consists of an arena $\mathcal{A}$ and a coloring function $c : V \to \mathbb{N}$. Player 0 wins play π if* $\max\{c(q) \mid q \in In(\pi)\}$ *is even, otherwise Player 1 wins.*
- ...

Definition 6

- *A strategy f_σ is* p-winning *for player σ and position p if all plays that conform to f_σ and that start in p are won by Player σ.*
- *The* winning region *for player σ is the set of positions*

$$W_\sigma = \{p \in V \mid \textit{there is a strategy } f_\sigma \textit{ s.t. } f_\sigma \textit{ is } p\textit{-winning}\}.$$

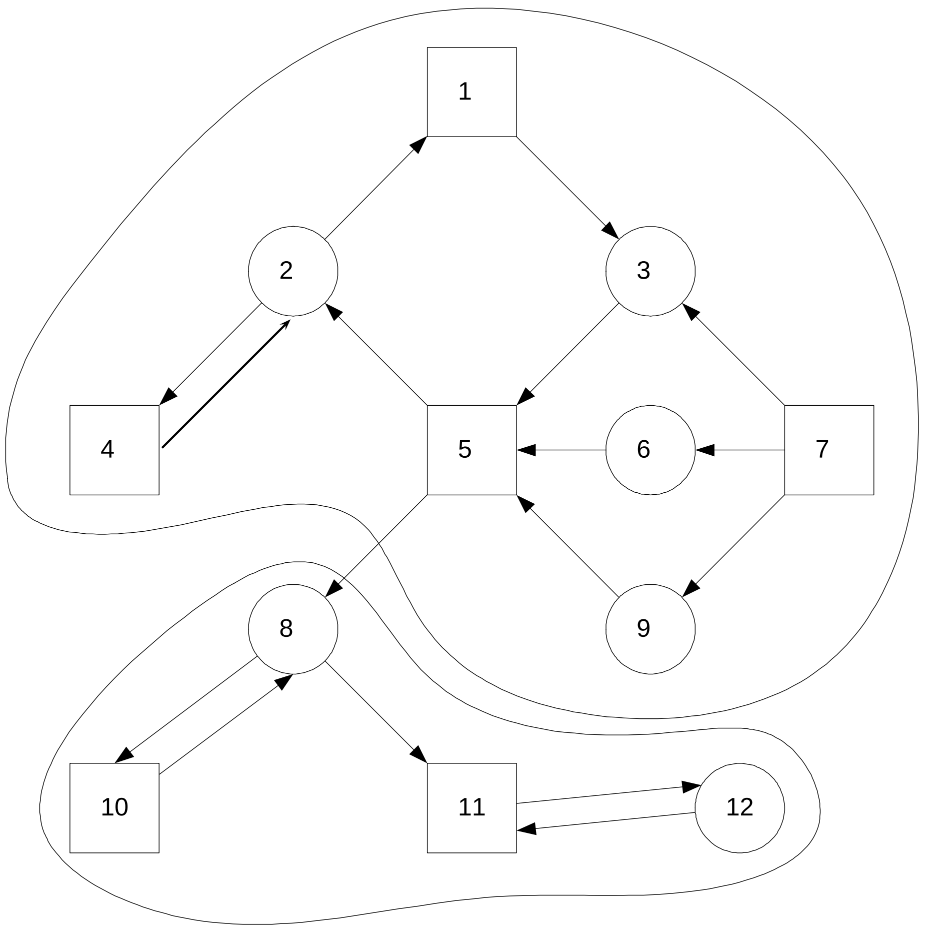
Definition 7 *A game is* determined *if $V = W_0 \cup W_1$.*

Definition 8

- *A* memoryless *strategy for player σ is a function $f_\sigma : V_\sigma \to V$ which defines a strategy $f'_\sigma(u \cdot v) = f(v)$.*
- *A game is* memoryless determined *if for every position some player wins the game with memoryless strategy.*

14 Solving Reachability Games

Example:



$\square$ = Player 0; $\bigcirc$ = Player 1;
$R = \{1,4\}$, $W_0 = \{1,2,3,4,5,6,7,9\}$, $W_1 = \{8,10,11,12\}$.

Attractor Construction:

$$
\begin{aligned}
Attr_\sigma^0(X) &= \emptyset;\\
Attr_\sigma^{i+1}(X) &= Attr_\sigma^i(X)\\
&\quad\cup\ \{p \in V_\sigma \mid \exists p' \,.\, (p,p') \in E \wedge p' \in Attr_\sigma^i(X) \cup X\}\\
&\quad\cup\ \{p \in V_{1-\sigma} \mid \forall p' \,.\, (p,p') \in E \Rightarrow p' \in Attr_\sigma^i(X) \cup X\};
\end{aligned}
$$

$$Attr_\sigma^+(X) = \bigcup_{i\in\omega} Attr_\sigma^i(X).$$

$$Attr_\sigma(X) = Attr_\sigma^+(X) \cup X$$

Theorem 1 *Reachability games are memoryless determined.*

Proof:

Let $q \in V$.

1. If $p \in Attr_0(R)$, then $p \in W_0$, with memoryless strategy f_0:
   - Fix an arbitrary total ordering on V.
   - for $p \in V_0$ we define $f_0(q)$:
     - if $p \in Attr_0^i(R)$ for some smallest $i > 0$, choose the minimal $p' \in Attr_0^{i-1}(R) \cup R$.
     - otherwise, choose the minimal $p' \in V$ such that $(p, p') \in E$.
   - Hence, if $p \in Attr_0^i(R)$ for some i, then any play that conforms to f_0 reaches R in at most i steps.
2. If $p \notin Attr_0(R)$, then $p \in W_1$ with memoryless strategy f_1:
   - for $p \in V_1$ we define $f_1(q)$:
     - if $p \in V_1 \setminus Attr_0(R)$, pick minimal $p' \in V \setminus Attr_0(R)$ such that $(p, p') \in E$. Such a p' must exist, since otherwise $p \in Attr_0(R)$.
     - otherwise, pick minimal $p' \in V$ such that $(p, p') \in E$.
   - Hence, if $p \in V \setminus Attr_0(R)$, then any play that conforms to f_1 never visits $Attr_0(R)$ and hence never R.

■

15 Solving Büchi Games

Recurrence Construction:

$$Recur_\sigma^0 = F;$$
$$Recur_\sigma^{i+1} = F \cap Attr_\sigma^+(Recur_\sigma^i);$$
$$Recur_\sigma = \bigcap_{i \in \omega} Recur_\sigma^i.$$

Theorem 2 *Büchi games are memoryless determined.*

Proof:

- If $p \in Attr_0(Recur_0)$, then $p \in W_0$, with memoryless strategy f_0:
  - Fix an arbitrary total ordering on V.
  - for $p \in V_0$ we define $f_0(q)$:
    - if $p \in Attr_0(Recur_0)$, choose
      - the minimal $p' \in Recur_0$, if $(p, p') \in E$ exists,
      - the minimal $p' \in Attr_0^i(Recur_0)$ for minimal i such that $(p, p') \in E$ exists, otherwise.

    * if $p \notin Attr_0(Recur_0)$, choose minimal $p' \in V$ with $(p, p') \in E$.

- If $p \notin Attr_0(Recur_0)$, then $p \in W_1$ with memoryless strategy f_1: we define memoryless strategies f_1^i such that if a play starts in $p \in V \setminus Attr_0^+(Recur_0^i)$ and conforms to f_1^i, then there are at most i further visits to F (not counting a possible visit in the first position).
  - $f_1^0(p)$: choose minimal $p' \in V$ such that $(p, p') \in E$ and $p' \in V \setminus Attr_0(F)$.
  - if $p \in V \setminus Attr_0^+(Recur_0^i)$, $f_1^{i+1}(p) = f_1^i(p)$;
  - if $p \notin V \setminus Attr_0^+(Recur_0^i)$, i.e., if $p \in Attr_0^+(Recur_0^i) \setminus Attr_0^+(Recur_0^{i+1})$, then for $f_1^{i+1}(p)$ choose minimal p' such that $(p, p') \in E$ and $p' \in Attr_0^+(Recur_0^i) \setminus Attr_0^+(Recur_0^{i+1})$.
- Induction on i:
  - $i = 0$: Player 1 can avoid $Attr_0(F)$ and hence F;
  - $i + 1$:
    * case 1: play never reaches F;
    * case 2: play reaches $p' \in F \setminus Recur_0^{i+1} = F \setminus Attr_0^+(Recur_0^i) \subseteq V \setminus Attr_0^+(Recur_0^i)$; by induction hypothesis, at most i further visits to F, not counting the visit in p', hence a total of at most $i + 1$ visits from p.

■

Bernd Finkbeiner
Sven Schewe

Date: July 10, 2008

Automata, Games and Verification: Lecture 11

16 Parity Games

Assumptions:

- arena is finite or countably infinite.
- the number of colors is finite (max color k).

Lemma 1 (Merging strategies) *Given a parity game $\mathcal{G}$ and a set of nodes $U \subseteq V$, s.t. for every $p \in U$, Player σ has a memoryless strategy $f_{\sigma,p}$ that wins from p, then there is a memoryless winning strategy f_σ that wins from all $p \in U$.*

Proof:

- Index the positions in $V = \{p_0, p_1, p_2, \ldots\}$
- For $p_i \in V$, let $F_i \subseteq V$ be the set of positions that are reachable from p_i in plays that conform to f_{p_i}.
- Define $f_\sigma(q) = f_{\sigma,p_i}(q)$ for the smallest i such that $q \in F_i$.
- f is winning for Player 0:
  - Applying f_σ corresponds to applying f_{σ,p_i} with weakly decreasing i.
  - From some point onward, $i = i^*$ is constant.
  - The play is won because $f_{\sigma,p_{i^*}}$ is winning.

■

Theorem 1 *Parity games are memoryless determined.*

Proof:

Induction on k:

- $k = 0$: $W_0 = V, W_1 = \emptyset$. Memoryless winning strategy: fix arbitrary order on V. $f_0(p) = \min\{q \mid (p,q) \in E\}$.
- $k + 1$:
  - If $k + 1$, consider player $\sigma = 0$, otherwise $\sigma = 1$.
  - Let $W_{1-\sigma}$ be the set of positions where Player $(1 - \sigma)$ has a memoryless winning strategy. We show that Player σ has a memoryless winning strategy from $V \setminus W_{1-\sigma}$.
  - Consider subgame $\mathcal{G}'$:

  * $V'_0 = V_0 \setminus W_{1-\sigma}$;
  * $V'_1 = V_1 \setminus W_{1-\sigma}$;
  * $E' = W \cap (V' \times V')$;
  * $c'(p) = c(p)$ for all $p \in V'$.
- $\mathcal{G}'$ is still a game:
  * for $p \in V'_\sigma$, there is a $q \in V \setminus W_{1-\sigma}$ with $(p, q) \in E'$, otherwise $p \in W_{1-\sigma}$;
  * for $p \in V'_{1-\sigma}$, for all $q \in V$ with $(p, q) \in E$, $q \in V \setminus W_{1-\sigma}$, hence there is a $q \in V'$ with $(p, q) \in E$.
- Let $C'_i = \{p \in V' \mid c'(p) = i\}$.
- Let $Y = Attr'_\sigma(C'_{k+1})$. ($Attr'$: Attractor set on $\mathcal{G}'$)
- Let f_A be the attractor strategy on $\mathcal{G}'$ into C'_{k+1}.
- Consider subgame $\mathcal{G}''$:
  * $V''_0 = V'_0 \setminus Y$;
  * $V''_1 = V_1 \setminus Y$;
  * $E' = W \cap (V'' \times V'')$;
  * $C'' : V'' \to \{0, \ldots, k\}; c''(p) = c'(p)$ for all $p \in V''$.
- $\mathcal{G}''$ is still a game.
- Induction hypothesis: $\mathcal{G}''$ is memoryless determined.
- Also: $W''_{1-\sigma} = \emptyset$ (because $W''_{1-\sigma} \subseteq W_{1-\sigma}$: assume Player $(1-\sigma)$ had a winning strategy from some position in V''. Then this strategy would win in $\mathcal{G}$, too, since Player σ has no chance to leave $\mathcal{G}''$ other than to $W_{1-\sigma}$.)
- Hence, there is a winning memoryless winning strategy f_{IH} for player σ from V''.
- We define:

$$f_\sigma(p) = \begin{cases} f_{IH}(p) & \text{if } p \in V''; \\ f_A(p) & \text{if } p \in Y \setminus C'_{k+1}; \\ \text{min. successor in } V \setminus W_{1-\sigma} & \text{if } p \in Y \cap C'_{k+1}; \\ \text{min. successor in } V & \text{otherwise.} \end{cases}$$

- f_σ is winning for Player σ on $V \setminus W_{1-\sigma}$.
  Consider a play that conforms to f_σ:
  * Case 1: Y is visited infinitely often.
    $\Rightarrow$ Player σ wins (inf. often even color $k+1$).
  * Case 2: Eventually only positions in V'' are visited.
    $\Rightarrow$ Since Player σ follows f_{IH}, Player σ wins.

$\blacksquare$

17 Tree Automata

Binary Tree: $T = \{0,1\}^*$.
Notation: T_Σ : set of all binary Σ-trees

Definition 1 *A tree automaton (over binary Σ-trees) is a tuple $\mathcal{A} = (S, s_0, M, \varphi)$:*

- *S: finite set of states*
- *$s_0 \in S$*
- *$M = S \times \Sigma \times S \times S$*
- *φ: acceptance condition (Büchi, parity, ...)*

Definition 2 *A run of a tree automaton $\mathcal{A}$ on a Σ-tree v is a S-tree (T, r), s.t.*

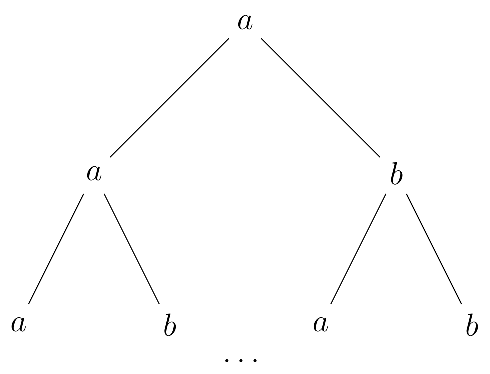
- *$r(\epsilon) = s_0$*
- *$(r(q), v(q), r(q0), r(q1)) \in M$ for all $q \in \{0,1\}^*$*

Definition 3 *A run is accepting if every branch is accepting (by φ). A Σ-tree is accepted if there exists an accepting run.*
$\mathcal{L}(A) :=$ set of accepted Σ-trees.

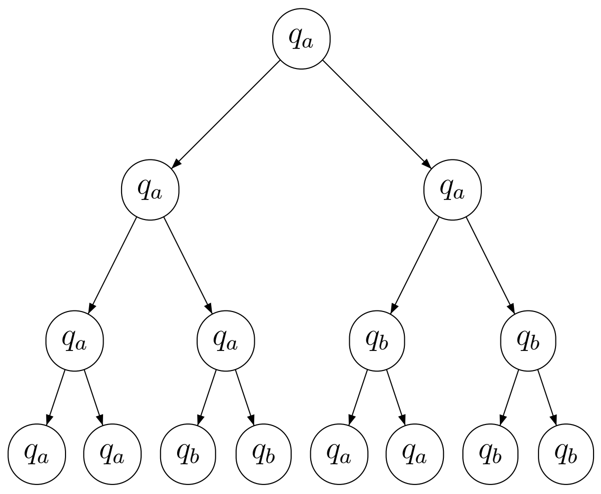
Example: $\{a, b\}$-trees with infinitely many bs on each path.

$\mathcal{A} = (S, s_0, M, c); \Sigma = \{a, b\}$;
$S = \{q_a, q_b\}; s_0 = q_a$;
$M = \{(q_a, a, q_a, q_a), (q_b, a, q_a, q_a), (q_a, b, q_b, q_b), (q_a, a, q_b, q_b), \ldots\}$;
Büchi $F = \{q_b\}$.

Σ-tree:



run:



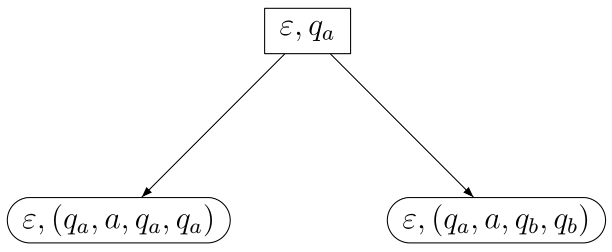
...

✚

Theorem 2 *A parity tree automaton $\mathcal{A} = (S, s_0, M, c)$ accepts an input tree t iff Player 0 wins the parity game $\mathcal{G}_{\mathcal{A},t} = (V_0, V_1, E, c')$ from position (ε, s_0).*

- $V_0 = \{(w, q) \mid w \in \{0,1\}^*, q \in S\}$;
- $V_1 = \{(w, \tau) \mid w \in \{0,1\}^*, \tau \in M\}$;
- $E = \{((w,q),(w,\tau)) \mid \tau = (q, t(w), q_0', q_1'), \tau \in M\}$
  $\cup \{((w,\tau),(w',q')) \mid \tau = (q, \sigma, q_0', q_1')$ *and*
  $((w' = w0$ *and* $q' = q_0')$ *or* $(w' = w1$ *and* $q' = q_1'))\}$;
- $c'(w, q) = c(q)$ *if* $q \in S$;
- $c'(w, \tau) = 0$ *if* $\tau \in M$.

Example:


ε, q_a
$\varepsilon, (q_a, a, q_a, q_a)$
$\varepsilon, (q_a, a, q_b, q_b)$

...

✚

Proof:

- Given an accepting run r construct a winning strategy f_0:

$$f_0(w,q) = (w,(r(w),t(w),r(w0),r(w1))$$

- Given a memoryless winning strategy f_0 construct an accepting run $r(\varepsilon) = s_0$ $\forall w \in \{0,1\}^*$
  - $r(w0) = q$ where $f_0(w,r(w)) = (w,(\_,\_,q,\_))$
  - $r(w1) = q$ where $f_0(w,r(w)) = (w,(\_,\_,\_,q))$

■

Lemma 2 *For each parity tree automaton $\mathcal{A}$ over Σ-trees there exists a parity tree automaton $\mathcal{A}'$ over $\{1\}$-trees, such that $\mathcal{L}(\mathcal{A}) = \emptyset$ iff $\mathcal{L}(\mathcal{A}') = \emptyset$.*

Proof:

- $S' = S$;
- $s'_0 = s_0$;
- $M' = \{(q,1,q_0.q_1) \mid (q,\sigma,q_0,q_1) \in M, \sigma \in \Sigma\}$
- $c' = c$

■

Theorem 3 *The language of a parity tree automaton $\mathcal{A} = (S,s_0,M,c)$ is non-empty iff Player 0 wins the parity game $\mathcal{G}_{\mathcal{A},t} = (V_0,V_1,E,c')$ from position s_0.*

- *$V_0 = S$;*
- *$V_1 = M$;*
- $$\begin{aligned} E = &\{(q,\tau) \mid \tau = (q,1,q'_0,q'_1), \tau \in M\} \\ &\cup \{(\tau,q') \mid \tau = (q,1,q'_0,q'_1) \textit{ and} \\ &\qquad (q' = q'_0 \textit{ or } q' = q'_1)\}; \end{aligned}$$
- *$c'(q) = c(q)$ for $q \in S$;*
- *$c(\tau) = 0$ for $\tau \in M$.*

Bernd Finkbeiner
Sven Schewe

Date: July 17, 2008

Automata, Games and Verification: Lecture 12

18 Complementation of Parity Tree Automata

Reference: W. Thomas: *Languages, Automata and Logic,* Handbook of formal languages, Volume 3.

Theorem 1 *For each parity tree automaton $\mathcal{A}$ over Σ there is a parity tree automaton $\mathcal{A}'$ with $\mathcal{L}(\mathcal{A}') = T_\Sigma - \mathcal{L}(\mathcal{A})$.*

Proof:

- $\mathcal{A}$ does *not* accept some tree t iff Player 1 has a winning memoryless strategy f in $\mathcal{G}_{\mathcal{A},t}$ from (ε, s_0)
- Strategy
$$f : \{0,1\}^* \times M \rightarrow \{0,1\}^* \times S$$
can be represented as
$$f' : \{0,1\}^* \times M \rightarrow \{0,1\}$$
(where $f(u,(q,\sigma,q_0',q_1')) = (u \cdot i, q_i')$ iff $f'(u,\tau) = i$).
- f' is isomorphic to
$$g : \{0,1\}^* \rightarrow (M \rightarrow \{0,1\})$$
($M \rightarrow \{0,1\}$ is the finite "local strategy")
- Hence, $\mathcal{A}$ does not accept t iff

  (1) there is a $(M \rightarrow \{0,1\})$-tree v such that
    (2) for all $i_0, i_1, i_2, \ldots \in \{0,1\}^\omega$
      (3) for all $\tau_0, \tau_1, \ldots \in M^\omega$
        (4) if
          – for all j,
            $\tau_j = (q, a, q_0', q_1')$
            $\Rightarrow a = t(i_0, i_1, \ldots, i_j)$ and
          – $i_0 i_1 \ldots = v(\varepsilon)(\tau_0) v(i_0)(\tau_1) \ldots$

        then the generated state sequence $q_0 q_1 \ldots$
          with $q_0 = s_0, (q_j, a, q_0', q_1') = \tau_j$,
          $q_{j+1} = q_{v(i_1,\ldots,i_j)(\tau_j)}$
          violates c.

- Condition **(4)** is a property of words over

$$\Sigma' = \underbrace{(M \to \{0,1\})}_{v} \times \underbrace{\Sigma}_{t} \times \underbrace{M}_{\tau} \times \underbrace{\{0,1\}}_{i}$$

  and can be checked by a parity word automaton $\mathcal{A}_4 = (S_4, \{s_4\}, T_4, c_4)$:
  - $S_4 = S \cup \{\bot\}$;
  - $s_4 = s_0$;
  - $T_4 = \{(q,(f,a,(q,a,q'_0,q'_1),i),q'_i) \mid q \in S, f : M \to \{0,1\},$
    $(q,a,q'_0,q'_1) \in M, i = f(q,a,q'_0,q'_1)\}$
    $\cup\ \{(q,(f,a,(q,a',q'_0,q'_1),i),\bot) \mid a \neq a' \text{ or } i \neq f(q,a',q'_0,q'_1)\}$
    $\cup\ \{(\bot,a,\bot) \mid a \in \Sigma'\}$;
  - $c_4(q) = c(q) + 1$ for $q \in S$;
  - $c_4(\bot) = 0$.
- Condition **(3)** is a property of words $(M \to \{0,1\}) \times \Sigma \times \{0,1\}$ which results from **(4)** by universal quantification (= complement; project; complement) $\Rightarrow$ there is a deterministic parity word automaton $\mathcal{A}_3$ that checks **(3)**.
- Condition **(2)** defines a property of $(M \to \{0,1\}) \times \Sigma$-trees. It can be checked by a tree automaton $\mathcal{A}_2 = (S_2, s_2, M_2, c_2)$, simulating $\mathcal{A}_3$ along each path:
  - $S_2 = S_3$;
  - $s_2 = s_3$;
  - $M_2 = \{(q,(f,a),q'_0,q'_1) \mid (q,(f,a,0),q'_0) \in T_3, (q,(f,a,1),q'_1) \in T_3\}$;
  - $c_2 = c_1$.
- Condition **(1)** is a property on Σ-trees: Use nondeterminism to guess $M \to \{0,1\}$ label: $\mathcal{A}_1 = (S_1, s_1, M_1, c_1)$, where
  - $S_1 = S_2$;
  - $s_1 = s_2$;
  - $M_1 = \{(q,a,q'_0,q'_1) \mid \exists f : M \to \{0,1\}.(q,(f,a),q'_0,q'_1) \in M_2\}$;
  - $c_1 = c_2$.

■

19 Monadic Second-Order Theory of Two Successors (S2S)

Syntax:

- first-order variable set $V_1 = \{x_0, x_1, \ldots\}$
- second-order variable set $V_2 = \{X_0, X_1, \ldots\}$
- Terms t:

$$t ::= \epsilon \mid x \mid t0 \mid t1$$

- Formulas φ:

$$\varphi ::= t \in X \mid t_1 = t_2 \mid \neg\varphi \mid \varphi_0 \vee \varphi_1 \mid \exists x.\varphi \mid \exists X.\varphi$$

Semantics:

- first-order valuation $\sigma_1 : V_1 \to \mathbb{B}^*$
- second-order valuation $\sigma_2 : V_2 \to 2^{\mathbb{B}^*}$

Semantics of terms:

- $[\![\epsilon]\!] = \epsilon$
- $[\![x]\!]_{\sigma_1} = \sigma_1(x)$
- $[\![t0]\!]_{\sigma_1} = [\![t]\!]_{\sigma_1}0$
- $[\![t1]\!]_{\sigma_1} = [\![t]\!]_{\sigma_1}1$

Semantics of formulas:

- $\sigma_1, \sigma_2 \models t \in X$ iff $[\![t]\!]_{\sigma_1} \in \sigma_2(X)$
- $\sigma_1, \sigma_2 \models t_1 = t_2$ iff $[\![t_1]\!]_{\sigma_1} = [\![t_2]\!]_{\sigma_1}$
- $\sigma_1, \sigma_2 \models \neg\varphi$ iff $\sigma_1, \sigma_2 \not\models \varphi$
- $\sigma_1, \sigma_2 \models \varphi_0 \vee \varphi_1$ iff $\sigma_1, \sigma_2 \models \varphi_0$ or $\sigma_1, \sigma_2 \models \varphi_1$
- $\sigma_1, \sigma_2 \models \exists x_i.\varphi$ iff there is a $a \in \mathbb{B}^*$ s.t.

$$\sigma_1'(y) = \begin{cases} \sigma_1(y) & \text{if} x \neq y, \\ a & \text{otherwise;} \end{cases}$$

  and $\sigma_1', \sigma_2 \models \varphi$
- $\sigma_1, \sigma_2 \models \exists X_i.\varphi$ iff there is a $A \subseteq \mathbb{B}^*$ s.t.

$$\sigma_2'(Y) = \begin{cases} \sigma_2(Y) & \text{if} X \neq Y \\ A & \text{otherwise;} \end{cases}$$

  and $\sigma_1, \sigma_2' \models \varphi$

Examples:

- "node x is a prefix of node y"

  $x \leqslant y \quad \Leftrightarrow \quad \forall X.((y \in X \wedge \forall z(z0 \in X \Rightarrow z \in X) \wedge \forall z.(z1 \in X \Rightarrow z \in X)) \Rightarrow x \in X)$

- "X is linearly ordered by $\leqslant$"

$$\text{Chain}(X) \quad \Leftrightarrow \quad \forall x.\forall y.((x \in X \wedge y \in X) \Rightarrow (x \leqslant y \vee y \leqslant x))$$

- "X is a path"

$$\begin{array}{rcl} \mathrm{Path}(X) & \Leftrightarrow & \mathrm{Chain}(X) \wedge \neg\exists Y.\ (X \subseteq Y \wedge X \neq Y \wedge \mathrm{Chain}(Y)) \\ X \subseteq Y & \Leftrightarrow & \forall z.(z \in X \Rightarrow z \in Y) \\ X = Y & \Leftrightarrow & X \subseteq Y \wedge Y \subseteq X \end{array}$$

Theorem 2 *For each Muller tree automaton $\mathcal{A} = (S, s_0, M, \mathcal{F})$ over $\Sigma = 2^{V_2}$ there is a S2S formula φ over V_2 s.t. $t \in \mathcal{L}(\mathcal{A})$ iff $\sigma_2 \models \varphi$ where $\sigma_2(P) = \{q \in \{0,1\}^* \mid P \in t(q)\}$.*

Theorem 3 *For every S2S formula φ over V_1, V_2 there is a Muller tree automaton $\mathcal{A}$ over $\Sigma = 2^{V_1 \cup V_2}$ such that $t \in \mathcal{L}(\mathcal{A})$ iff $\sigma_1, \sigma_2 \models \varphi$ where*

$$\begin{array}{rcl} \sigma_1(x) & = & q \text{ iff } x \in t(q); \\ \sigma_2(X) & = & \{q \in \{0,1\}^* \mid X \in t(q)\}. \end{array}$$

Theorem 4 *S2S is decidable.*

SnS is the monadic second order theory of n successors.

Theorem 5 *SnS is decidable.*

20 Synthesis

The Synthesis Problem: Let i be a Boolean input variable, and O be a set of Boolean output variables. Given an LTL specification φ over $O \cup \{i\}$, decide if *there exists* an implementation that satisfies φ for all possible inputs.

Construction:

$$\begin{tikzcd}
 & \textit{LTL specification } \varphi \arrow[d] & \\
 & \text{Alternating Büchi word automaton } \mathcal{A}_\varphi \arrow[d] & \\
 & \text{Nondeterministic Büchi word automaton } \mathcal{A}'_\varphi \arrow[d] & \\
 & \text{Deterministic parity word automaton } \mathcal{A}''_\varphi \arrow[d] & \\
 & \text{Deterministic parity } \textit{tree} \text{ automaton } \mathcal{A}'''_\varphi \arrow[d] & \\
 & \text{Empty?} \arrow[dl] \arrow[dr] & \\
\text{Yes: } \varphi \textit{ not realizable} & & \begin{array}{c}\text{No: } \varphi \textit{ realizable} \\ \text{winning strategy} \\ \text{in emptiness game} \\ \text{defines implementation.}\end{array}
\end{tikzcd}$$