Büchi Automata

Lecture #9 of Model Checking

Joost-Pieter Katoen

Lehrstuhl 2: Software Modeling & Verification

E-mail: katoen@cs.rwth-aachen.de

Mai 2, 2007

Overview Lecture #9

$\Rightarrow$ Motivation: Peterson's Algorithm

- ω-Regular Languages
- Nondeterministic Büchi Automata (NBA)
- NBA and ω-Regular Languages

Peterson's banking system

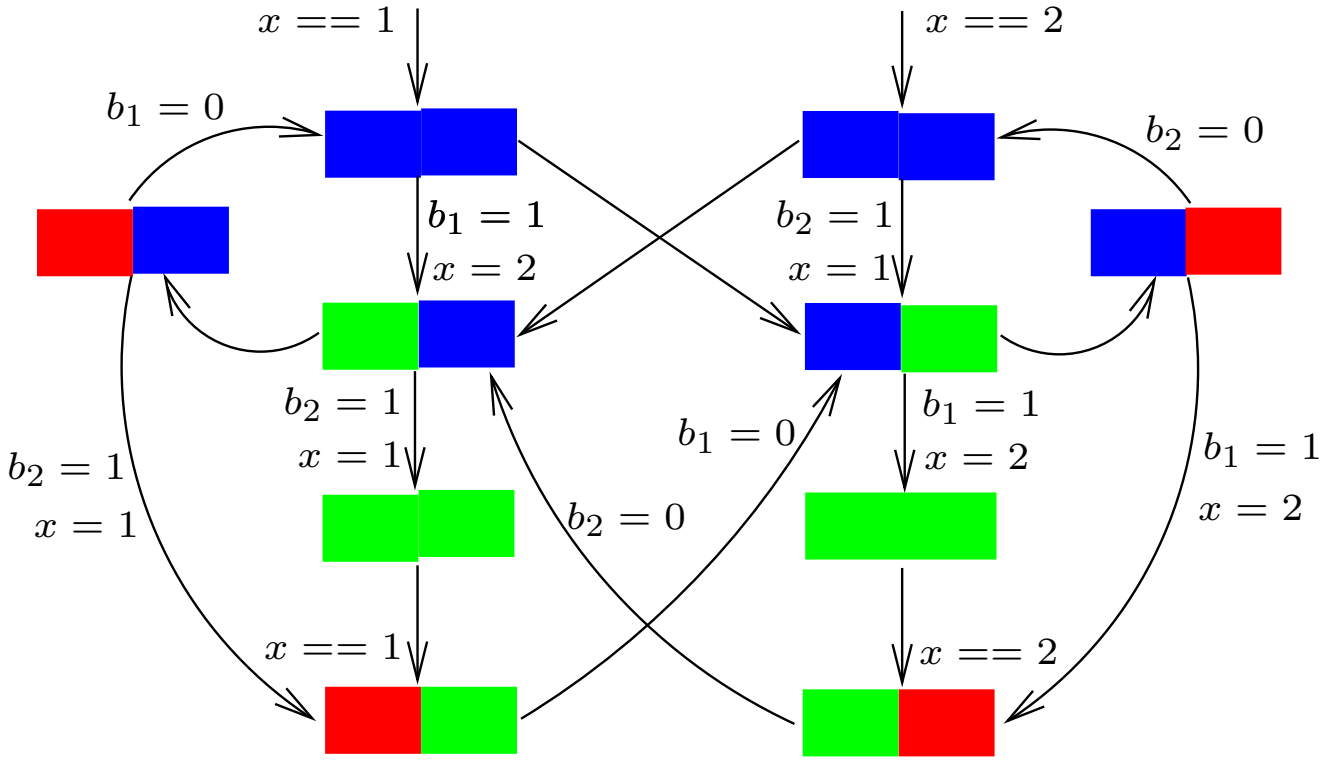
Person Left behaves as follows:

```
          while true {
              ......
rq :          b1, x = true, 2;
wt :          wait until(x == 1 || ¬ b2) {
cs :              ... @account_L ...}
              b1 = false;
              ......
          }
```

Person Right behaves as follows:

```
          while true {
              ......
rq :          b2, x = true, 1;
wt :          wait until(x == 2 || ¬ b1) {
cs :              ... @account_R ...}
              b2 = false;
              ......
          }
```

Is the banking system live?

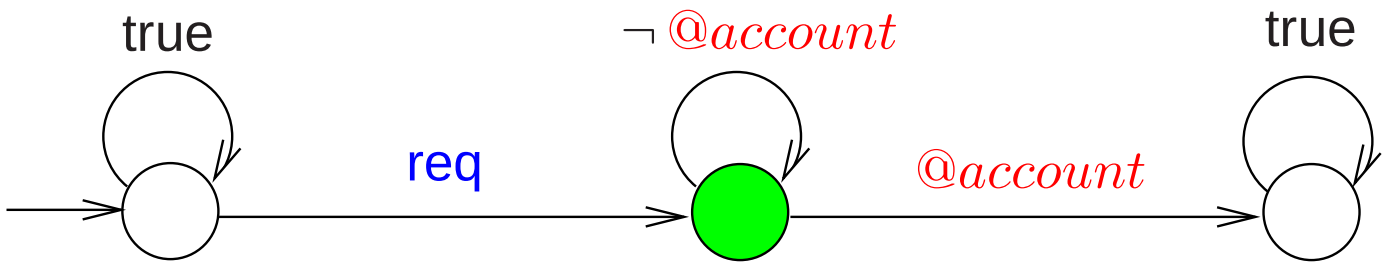


If someone wants to update the account, does he ever get the opportunity to do so?

"always ($req_L \Rightarrow$ eventually @account$_L$) $\wedge$ always ($req_R \Rightarrow$ eventually @account$_R$)"

Is the banking system live (revisited)?

- Live = when you want access to the account, you eventually get it
- Unlive: once you want access to the account, you never get it
  - unlive behaviour can be characterized as a (set of) infinite traces
  - or, equivalently, by a Büchi-automaton:



- Checking liveness: $\mathit{Traces}(\mathit{System}) \cap L_\omega(\overline{\mathit{Live}}) = \varnothing$?
  - (explicit) complementation, intersection and emptiness of Büchi automata!

ω-regular expressions

1. $\underline{\varnothing}$ and $\underline{\varepsilon}$ are regular expressions over Σ

2. if $A \in \Sigma$ then $\underline{A}$ is a regular expression over Σ

3. if E, E_1 and E_2 are regular expressions over Σ
   then so are $\mathsf{E}_1 + \mathsf{E}_2$, $\mathsf{E}_1.\mathsf{E}_2$ and E^*

E^+ is an abbreviation for the regular expression $\mathsf{E}.\mathsf{E}^*$

An ω-*regular expression* G over the alphabet Σ has the form:

$$\mathsf{G} = \mathsf{E}_1.\mathsf{F}_1^\omega + \ldots + \mathsf{E}_n.\mathsf{F}_n^\omega \quad \text{for } n > 0$$

where E_i, F_i are regular expressions over Σ such that $\varepsilon \notin \mathcal{L}(\mathsf{F}_i)$, for all $0 < i \leqslant n$

Semantics of ω-regular expressions

- The *semantics* of regular expression E is a language $\mathcal{L}(\mathsf{E}) \subseteq \Sigma^*$:

$$\mathcal{L}(\underline{\varnothing}) = \varnothing, \quad \mathcal{L}(\underline{\varepsilon}) = \{\,\varepsilon\,\}, \quad \mathcal{L}(\underline{A}) = \{\,A\,\}$$

$$\mathcal{L}(\mathsf{E}{+}\mathsf{E}') = \mathcal{L}(\mathsf{E}) \cup \mathcal{L}(\mathsf{E}') \quad \mathcal{L}(\mathsf{E}.\mathsf{E}') = \mathcal{L}(\mathsf{E}).\mathcal{L}(\mathsf{E}') \quad \mathcal{L}(\mathsf{E}^*) = \mathcal{L}(\mathsf{E})^*$$

- The *semantics* of ω-regular expression G is a language $\mathcal{L}(\mathsf{G}) \subseteq \Sigma^\omega$:

$$\mathcal{L}_\omega(\mathsf{G}) \;=\; \mathcal{L}(\mathsf{E}_1).\mathcal{L}(\mathsf{F}_1)^\omega \cup \ldots \cup \mathcal{L}(\mathsf{E}_n).\mathcal{L}(\mathsf{F}_n)^\omega$$

- G_1 and G_2 are *equivalent*, denoted $\mathsf{G}_1 \equiv \mathsf{G}_2$, if $\mathcal{L}_\omega(\mathsf{G}_1) = \mathcal{L}_\omega(\mathsf{G}_2)$

ω-regular languages and properties

- $\mathcal{L} \subseteq \Sigma^\omega$ is *ω-regular* if $\mathcal{L} = \mathcal{L}_\omega(\mathrm{G})$ for some ω-regular expression G (over Σ)

- ω-regular languages possess several closure properties
  - they are closed under union, intersection, and complementation
  - complementation is not treated here; we use a trick to avoid it

- LT property P over *AP* is called *ω-regular*

  if P is an ω-regular language over the alphabet 2^{AP}

all invariants and regular safety properties are ω-regular!

Examples

Büchi automata

- NFA (and DFA) are incapable of accepting infinite words
- Automata on infinite words
  - suited for accepting ω-regular languages
  - we consider nondeterministic Büchi automata (NBA)
- Accepting runs have to “check” the entire input word $\Rightarrow$ are infinite

  $\Rightarrow$ acceptance criteria for infinite runs are needed
- NBA are like NFA, but have a distinct *acceptance criterion*
  - one of the accept states must be visited infinitely often

Büchi automata

A *nondeterministic Büchi automaton* (NBA) $\mathcal{A}$ is a tuple $(Q, \Sigma, \delta, Q_0, F)$ where:

- Q is a finite set of states with $Q_0 \subseteq Q$ a set of initial states
- Σ is an alphabet
- $\delta : Q \times \Sigma \to 2^Q$ is a transition function
- $F \subseteq Q$ is a set of accept (or: final) states

The size of $\mathcal{A}$, denoted $|\mathcal{A}|$, is the number of states and transitions in $\mathcal{A}$:

$$|\mathcal{A}| \;=\; |Q| + \sum_{q \in Q} \sum_{A \in \Sigma} |\, \delta(q, A) \,|$$

An example NBA

Language of an NBA

- NBA $\mathcal{A} = (Q, \Sigma, \delta, Q_0, F)$ and word $\sigma = A_0 A_1 A_2 \ldots \in \Sigma^\omega$
- A *run* for σ in $\mathcal{A}$ is an infinite sequence $q_0\, q_1\, q_2 \ldots$ such that:
  - $q_0 \in Q_0$ and $q_i \xrightarrow{A_i} q_{i+1}$ for all $0 \leqslant i$
- Run $q_0\, q_1\, q_2 \ldots$ is *accepting* if $q_i \in F$ for infinitely many i
- $\sigma \in \Sigma^\omega$ is *accepted* by $\mathcal{A}$ if there exists an accepting run for σ
- The *accepted language* of $\mathcal{A}$:

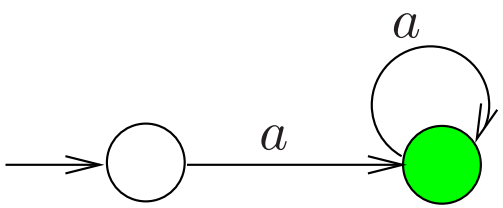
$$\mathcal{L}_\omega(\mathcal{A}) = \{\sigma \in \Sigma^\omega \mid \text{ there exists an accepting run for } \sigma \text{ in } \mathcal{A}\}$$

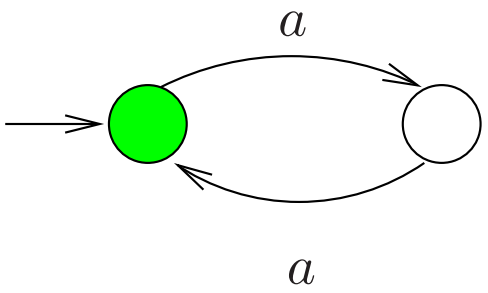
- NBA $\mathcal{A}$ and $\mathcal{A}'$ are *equivalent* if $\mathcal{L}_\omega(\mathcal{A}) = \mathcal{L}_\omega(\mathcal{A}')$

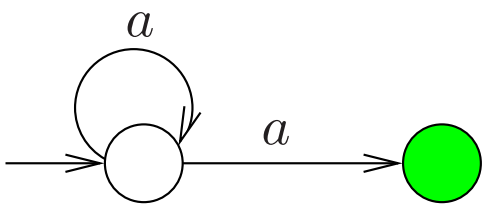
Example runs and accepted words

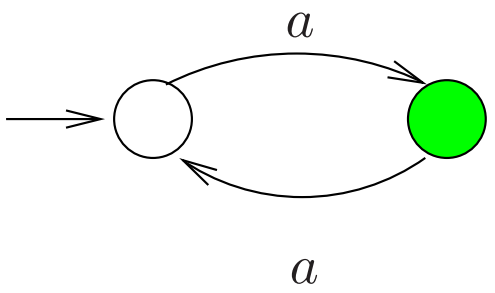
NBA and properties

NBA versus NFA









finite equivalence $\not\Rightarrow$ ω-equivalence

$\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$, but $\mathcal{L}_\omega(\mathcal{A}) \neq \mathcal{L}_\omega(\mathcal{A}')$

ω-equivalence $\not\Rightarrow$ finite equivalence

$\mathcal{L}_\omega(\mathcal{A}) = \mathcal{L}_\omega(\mathcal{A}')$, but $\mathcal{L}(\mathcal{A}) \neq \mathcal{L}(\mathcal{A}')$

NBA and ω-regular languages

The class of languages accepted by NBA

agrees with the class of ω-regular languages

(1) any ω-regular language is recognized by an NBA

(2) for any NBA $\mathcal{A}$, the language $\mathcal{L}_\omega(\mathcal{A})$ is ω-regular

For any ω-regular language there is an NBA

- How to construct an NBA for the ω-regular expression:

$$\mathsf{G} = \mathsf{E}_1.\mathsf{F}_1^\omega + \ldots + \mathsf{E}_n.\mathsf{F}_n^\omega \; ?$$

where E_i and F_i are regular expressions over alphabet Σ; $\varepsilon \notin \mathsf{F}_i$

- Rely on operations for NBA that mimic operations on ω-regular expressions:

(1) for NBA $\mathcal{A}_1$ and $\mathcal{A}_2$ there is an NBA accepting $\mathcal{L}_\omega(\mathcal{A}_1) \cup \mathcal{L}_\omega(\mathcal{A}_2)$
(2) for any regular language $\mathcal{L}$ with $\varepsilon \notin \mathcal{L}$ there is an NBA accepting $\mathcal{L}^\omega$
(3) for regular language $\mathcal{L}$ and NBA $\mathcal{A}'$ there is an NBA accepting $\mathcal{L}.\mathcal{L}_\omega(\mathcal{A}')$

Union of NBA

For NBA $\mathcal{A}_1$ and $\mathcal{A}_2$ (both over the alphabet Σ)

there exists an NBA $\mathcal{A}$ such that:

$$\mathcal{L}_\omega(\mathcal{A}) = \mathcal{L}_\omega(\mathcal{A}_1) \cup \mathcal{L}_\omega(\mathcal{A}_2) \quad \text{and} \quad |\mathcal{A}| = \mathcal{O}(|\mathcal{A}_1| + |\mathcal{A}_2|)$$

ω-operator for NFA

For each NFA $\mathcal{A}$ with $\varepsilon \notin \mathcal{L}(\mathcal{A})$ there exists an NBA $\mathcal{A}'$ such that:

$$\mathcal{L}_\omega(\mathcal{A}') = \mathcal{L}(\mathcal{A})^\omega \quad \text{and} \quad |\mathcal{A}'| = \mathcal{O}(|\mathcal{A}|)$$

Concatenation of an NFA and an NBA

For NFA $\mathcal{A}$ and NBA $\mathcal{A}'$ (both over the alphabet Σ

there exists an NBA $\mathcal{A}''$ with

$$\mathcal{L}_\omega(\mathcal{A}'') = \mathcal{L}(\mathcal{A}).\mathcal{L}_\omega(\mathcal{A}') \quad \text{and} \quad |\mathcal{A}''| = \mathcal{O}(|\mathcal{A}| + |\mathcal{A}'|)$$

Summarizing the results so far

For any ω-regular language $\mathcal{L}$

there exists an NBA $\mathcal{A}$ with $\mathcal{L}_\omega(\mathcal{A}) = \mathcal{L}$

NBA accept ω-regular languages

For each NBA $\mathcal{A}$: $\mathcal{L}_\omega(\mathcal{A})$ is ω-regular

Example