LTL to Büchi automata

Franco Raimondi

Department of Computer Science
School of Science and Technology
Middlesex University
http://www.rmnd.net

LTL quick recap

LTL Syntax (BNF)

$$\phi ::= \top \mid p \mid \phi \wedge \phi \mid \neg\phi \mid X\phi \mid \phi U \phi$$

Derived operators:

- $F\phi \equiv \top U \phi$
- $G\phi \equiv \neg \Diamond \neg \phi$

Intuition

Given a formula ϕ, let
$\text{Words}(\phi) = \{\sigma \in (2^{AP})^\omega | \sigma \models \phi\}$

- $TS \models \phi$ iff $\text{Traces}(TS) \subseteq \text{Words}(\phi)$
- iff $\text{Traces}(TS) \cap ((2^{AP})^\omega \setminus \text{Words}(\phi)) = \emptyset$
- iff $\text{Traces}(TS) \cap \text{Words}(\neg\phi)) = \emptyset$.

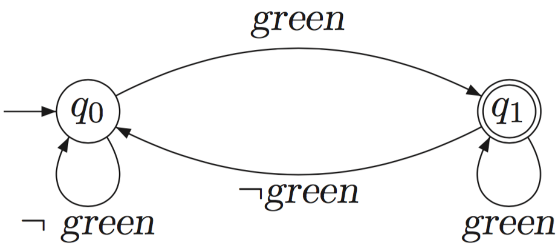
Suppose now that you have an automaton $\mathcal{A}$ s.t.
$\mathcal{L}(\mathcal{A}) = \text{Words}(\neg\phi)$.
Then $TS \models \phi$ iff $\text{Traces}(TS) \cap \mathcal{L}(\mathcal{A}) = \emptyset$.

Automata-based LTL model checking algorithm: intuition

- Given a formula ϕ and a TS.
- Build the automaton $\mathcal{L}(\mathcal{A})$ for $\neg\phi$.
- Construct the product automaton $TS \otimes \mathcal{L}$.
- If the product automaton is *not* empty, return NO (and an accepted word is a counterexample).
- Otherwise, return YES.

Example

The key point is building the automaton $\mathcal{A}$ for $\neg\phi$.
Example for $GF\,\mathrm{green}$:



Example 2

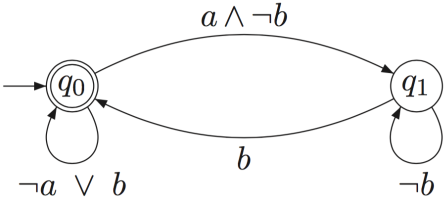


Figure 5.18: NBA for $\Box(a \rightarrow \Diamond b)$.

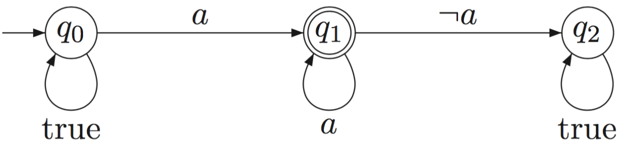


Figure 5.19: NBA for $\Diamond\Box a$.

Recap: release operator and PNF

- $\phi R \psi$: intuitively, ψ remains true if ψ does not become true.
- Important equivalence: $\phi R \psi \equiv \neg(\neg\phi U \neg\psi)$
- $G\phi \equiv \bot R \phi$ (and $\equiv \phi W \bot$, see above)

The algorithm to convert LTL to Büchi assumes positive normal form:

- $\neg(\phi \wedge \psi) \equiv \neg\phi \wedge \neg\psi$
- $\neg X\phi \equiv X\neg\phi$
- $\neg(\phi U \psi) \equiv \neg\phi R \neg\psi$

Recap: Büchi automaton

The following material is taken from: D. Giannakopoulou and F. Lerda, *From States to Transitions: Improving translation of LTL formulae to Büchi automata*, FORTE 2002.

- $B =< S, A, \Delta, q_0, F >$ where S is a finite set of states, A is a finite set of labels, $\Delta \subseteq S \times A \times S$ a transition relation, $q_0 \in S$ is the initial state and $F \subseteq S$ is a set of accepting states.
- An execution on an infinite word $w = a_0 a_1 \ldots$ is an infinite sequence of states $\sigma = s_0 s_1 \ldots$ s.t. $s_0 = q_0$ and $\forall i \geq 0, (s_i, a_i, s_{i+i}) \in \Delta$.
- A word w is *accepted* iff there exists an execution s.t. *some* element of F appears infinitely often in w

- $GBA = < S, A, \Delta, q_0, \mathcal{F} >$ where everything is as above and $\Delta \subseteq S \times A \times S$.
- An execution is an infinite word $\sigma = (s_0, l_0, t_0), (s_1, l_1, t_1), \ldots$ of source, label, target, s.t. $s_0 = q_0$ and $\forall i \geq 0, l_i = a_i$ and $t_i = s_{i+1}$.
- A word w over A is accepted iff there exists an execution s.t. *some* element form *each* set in $\mathcal{F}$ occurs infinitely often.

Remember: DBA $\neq$ NBA, but GBA and NBA are equally expressive.

From LTL to NBA: intuition

- From ϕ, build generalized Büchi automaton $\mathcal{G}_\phi$ s.t. $\mathcal{L}(\mathcal{G}_\phi) =$ Words(ϕ).
- *Degeneralize* $\mathcal{G}_\phi$ and build a corresponding NBA $\mathcal{A}_\phi$ s.t. $\mathcal{L}(\mathcal{G}_\phi) = \mathcal{L}(\mathcal{A}_\phi)$.
- States of $\mathcal{G}_\phi$ are sets of subformulas of ϕ.

From LTL to NBA: a few more details

Given LTL formula ϕ, let $\mathcal{G}_\phi = (Q, 2^{AP}, \delta, Q_0, \mathcal{F})$ where:

- Q is the set of elementary (see def. 5.35 in the book if curious) sets or formulae $B \subseteq \mathrm{closure}(\phi)$.
- $Q_0 = \{B \in Q | \phi \in B\}$
- $\mathcal{F} = \{F_{\phi_1 U \phi_2} | \phi_1 U \phi_2 \in \mathrm{closure}(\phi)\}$ where $F_{\phi_1 U \phi_2} = \{B \in Q | \phi_1 U \phi_2 \notin B \text{ or } \phi_2 \in B\}$.
- For the definition of δ, if curious, see book or article for a slightly different approach.

LTL2Buchi: a tool to convert form LTL to Büchi automata.

- Based on algorithm described in D. Giannakopoulou and F. Lerda, *From States to Transitions: Improving translation of LTL formulae to Büchi automata*, FORTE 2002.
- Initially embedded in Java Pathfinder (very old version).
- Separated project by Ewgenij Starostin as part of a JPF Google Summer of Code project in 2008.
- `https://github.com/fraimondi/ltl2buchi`

LTL2Buchi

LTL2Buchi: a tool to convert form LTL to Büchi automata.

- Based on algorithm described in D. Giannakopoulou and F. Lerda, *From States to Transitions: Improving translation of LTL formulae to Büchi automata*, FORTE 2002.
- Initially embedded in Java Pathfinder (very old version).
- Separated project by Ewgenij Starostin as part of a JPF Google Summer of Code project in 2008.
- `https://github.com/fraimondi/ltl2buchi`

[Demo: let's try some formulae]