Model-checking LTL properties

Deepak D'Souza

Department of Computer Science and Automation
Indian Institute of Science, Bangalore.

2, 4, 7 February 2022

Outline of this lecture

Syntax and semantics of LTL

Syntax:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid X\varphi \mid \varphi U \varphi.$$

Semantics: Given an infinite sequence of states $w = s_0 s_1 \cdots$, and a position $i \in \{0, 1, \ldots\}$, we define the relation $w, i \models \varphi$ inductively as follows:

$$\begin{array}{lll} w, i \models p & \text{iff} & p \text{ holds true in } s_i. \\ w, i \models \neg\varphi & \text{iff} & w, i \not\models \varphi. \\ w, i \models \varphi \vee \psi & \text{iff} & w, i \models \varphi \text{ or } w, i \models \psi. \\ w, i \models X\varphi & \text{iff} & w, i+1 \models \varphi. \\ w, i \models \varphi U \psi & \text{iff} & \exists j: \; i \leq j, \; w, j \models \psi, \text{ and} \\ & & \forall k: i \leq k < j, \; w, k \models \varphi. \end{array}$$

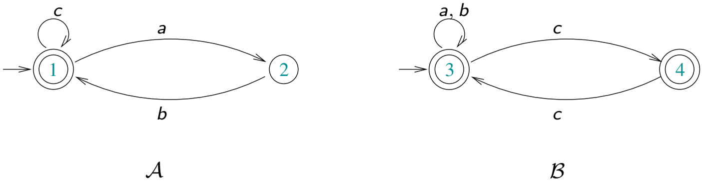
$F\varphi$ is shorthand for $true U \varphi$, and $G\varphi$ is shorthand for $\neg(F\neg\varphi)$.

When a system model satisfies an LTL property

If $\mathcal{T}$ is a transition system and φ is an LTL formula with propositions that refer to values of variables in $\mathcal{T}$, then we say $\mathcal{T} \models \varphi$ (read "$\mathcal{T}$ satisfies φ") iff each infinite execution of $\mathcal{T}$ satisfies φ in the initial position.

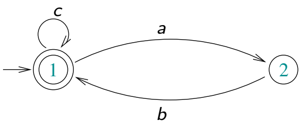
Model-Checking Algo: Idea

Can we give an algorithm to decide if $L(\mathcal{A}) \subseteq L(B)$?

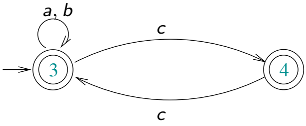


Model-Checking Algo: Idea

Can we give an algorithm to decide if $L(\mathcal{A}) \subseteq L(B)$?

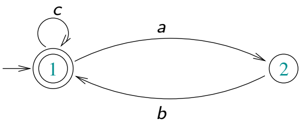


$\mathcal{A}$

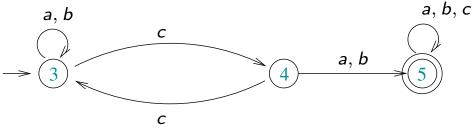


$\mathcal{B}$

First complement $\mathcal{B}$:



$\mathcal{A}$



$\overline{\mathcal{B}}$

Then construct the "product" of $\mathcal{A}$ and $\overline{\mathcal{B}}$, and check for emptiness.

Overview of LTL model-checking procedure

Given a transition system $\mathcal{T}$ and an LTL property φ, we want to know whether $\mathcal{T} \models \varphi$ (i.e. do all infinite executions of $\mathcal{T}$ satisfy φ?).

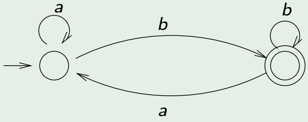
General idea:

- Compile given property φ into an automaton $\mathcal{A}_{\neg\varphi}$ accepting precisely the models of $\neg\varphi$.
- Take the "product" of $\mathcal{T}$ and $\mathcal{A}_{\neg\varphi}$.
- Look for an "accepting" path in this product.
- If such a path exists, this is a counter-example to the claim that $\mathcal{T}$ satisfies the property φ.
- If no such path exists, then $\mathcal{T}$ satisfies φ.

Büchi automata

- Finite state automata that run over infinite words.
  Example: $(ab)^\omega$ denotes the infinite string $abababab\cdots$.
- How do we accept an *infinite* word? Acceptance mechanism proposed by Büchi: see if run visits a final state infinitely often.

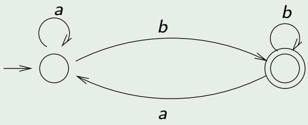
Büchi automaton for infinitely many b's



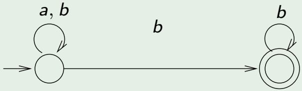
Büchi automata

- Finite state automata that run over infinite words.
  Example: $(ab)^\omega$ denotes the infinite string $ababababab \cdots$.
- How do we accept an *infinite* word? Acceptance mechanism proposed by Büchi: see if run visits a final state infinitely often.

Büchi automaton for infinitely many b's



Büchi automaton for finitely many a's



Exercise

Give a Büchi automaton over the alphabet $\{a, b, c\}$ that accepts all infinite strings in which every a is eventually followed by a b.

Checking non-emptiness of Büchi automata

- Büchi automata have similar closure properties to classical FSA's: closed under union, intersection, and complement.
- Non-emptiness is efficiently decidable: Look for a path from initial state to a final state that can reach itself.
- Can be checked efficiently: in time linear in the number of states and transitions of automaton.

Checking non-emptiness

Checking non-emptiness of Büchi automata

- Büchi automata have similar closure properties to classical FSA's: closed under union, intersection, and complement.
- Non-emptiness is efficiently decidable: Look for a path from initial state to a final state that can reach itself.
- Can be checked efficiently: in time linear in the number of states and transitions of automaton.

Checking non-emptiness

Generalized Büchi condition: $F_1, \ldots, F_k$ and a run is accepting if each F_i is seen infinitely often. Can be easily converted to a normal Büchi condition.

LTL models as sequences of propositional valuations

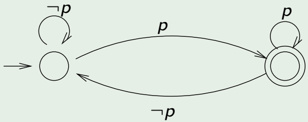
- LTL can be interpreted over a sequence of valuations to the propositions used in the formula.
    - E.g. In the formula `G((count = 1) => X(count = 2))`, `count = 1` and `count = 2` are the only propositions (say p and q), and a state can be viewed as a valuation to these propositions
- Example propositional valuation: $\langle p \mapsto \mathit{true}, \; q \mapsto \mathit{false} \rangle$.
- We represent such a valuation as simply $\{p\}$ (that is the subset of propositions that are true).
- Further use a propositional formula (like $p \vee q$) to represent *sets* of propositional valuations, namely those in which the formula is true.
    - E.g. $p \vee q$ represents the 3 valuations $\{p, q\}$, $\{p\}$, and $\{q\}$.

Compiling LTL properties into Büchi automata

Every LTL property φ over a set of propositions P can be expressed in the form of a BA $\mathcal{A}_\varphi$ over the the alphabet 2^P, that accepts precisely the models of φ.

Some examples over set of propositions $P = \{p, q\}$. The label "$\neg p$" is short for the set of labels $\{q\}$ and $\{\}$.

Büchi automaton for $G(F(p))$

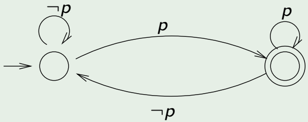


Compiling LTL properties into Büchi automata

Every LTL property φ over a set of propositions P can be expressed in the form of a BA $\mathcal{A}_\varphi$ over the the alphabet 2^P, that accepts precisely the models of φ.

Some examples over set of propositions $P = \{p, q\}$. The label "$\neg p$" is short for the set of labels $\{q\}$ and $\{\}$.

Büchi automaton for $G(F(p))$



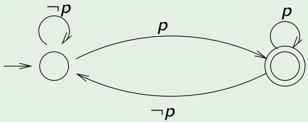
Büchi automaton for pUq

Compiling LTL properties into Büchi automata

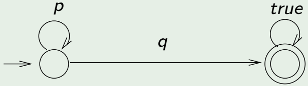
Every LTL property φ over a set of propositions P can be expressed in the form of a BA $\mathcal{A}_\varphi$ over the the alphabet 2^P, that accepts precisely the models of φ.

Some examples over set of propositions $P = \{p, q\}$. The label "$\neg p$" is short for the set of labels $\{q\}$ and $\{\}$.

Büchi automaton for $G(F(p))$

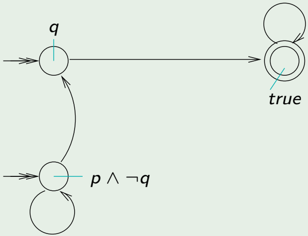


Büchi automaton for pUq



Automata with State-Based Labels

Example BA for pUq with labels on states



LTL to Büchi automata algorithmically

Step 1: Form the closure $cl(\varphi)$ of the given LTL formula φ as follows:

- Throw in all sub-formulas of φ.
- Throw in $X(\theta U \eta)$ whenever $\theta U \eta$ is a subformula.
- Throw in $\neg\psi$ for each ψ thrown in (identify $\neg\neg\theta$ with θ for this purpose).

Example: Closure of $pU\neg p$

$cl(pU\neg p) = \{p, \neg p, pU\neg p, X(pU\neg p), \neg X(pU\neg p), \neg(pU\neg p)\}$.

LTL to Büchi automata algorithmically

Step 2: Form "atoms" of given LTL formula φ. Atoms will play the role of states in the resulting BA.

- An atom of φ is a "maximally consistent" subset A of $cl(\varphi)$:
  - For each $\neg\psi \in cl(\varphi)$, A contains exactly one of ψ or $\neg\psi$.
  - For each $\theta \vee \psi \in cl(\varphi)$, A contains $\theta \vee \psi$ iff it contains θ or ψ.
  - For each $\theta U \eta \in cl(\varphi)$, A contains $\theta U \eta$ iff it contains η or both θ and $X(\theta U \eta)$.

Example: Atoms of $pU\neg p$

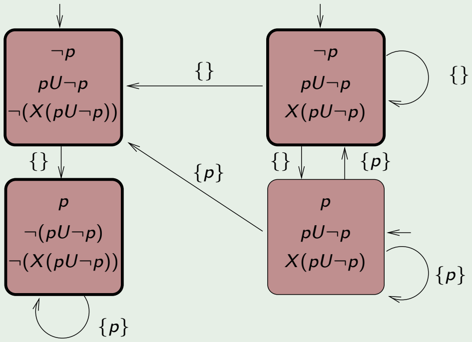
$\neg p$ $pU\neg p$ $\neg(X(pU\neg p))$	$\neg p$ $pU\neg p$ $X(pU\neg p)$
p $\neg(pU\neg p)$ $\neg(X(pU\neg p))$	p $pU\neg p$ $X(pU\neg p)$

LTL to Büchi automata algorithmically

Step 3: Add transition from atoms A to B labelled by $s \subseteq P$ if

- s is consistent with A (i.e. $s = A \cap P$).
- For each $X\psi$ in $cl(\varphi)$:
  - $X\psi \in A$ iff $\psi \in B$.

Example: formula atomaton for $pU\neg p$



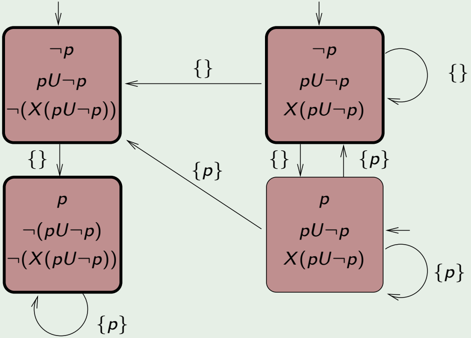
LTL to Büchi automata algorithmically

Step 4: Make atom A initial if A contains φ.

Step 5. Have a generalised Büchi acceptance: For each $\theta U \eta$ in $cl(\varphi)$:

$$F_{\theta U \eta} = \{A \mid \theta U \eta \notin A \text{ or } \eta \in A\}.$$

Example: Atomaton for $pU\neg p$. Final states $F_{pU\neg p}$ shown in bold.



Correctness of construction

Let $\mathcal{A}_\varphi$ be the Büchi automaton constructed by our algorithm. Then

Theorem

$L(\mathcal{A}_\varphi)$ accepts precisely the models of φ.

Correctness of construction

Let $\mathcal{A}_\varphi$ be the Büchi automaton constructed by our algorithm. Then

Theorem

$L(\mathcal{A}_\varphi)$ accepts precisely the models of φ.

Prove Soundess and Completeness of the formula automaton wrt the models of φ.

Model-checking LTL properties

Given a transition system $\mathcal{T}$ and an LTL property φ over a set of propositions P, we want to know whether $\mathcal{T} \models \varphi$ (i.e. do all infinite executions of $\mathcal{T}$ satisfy φ?).

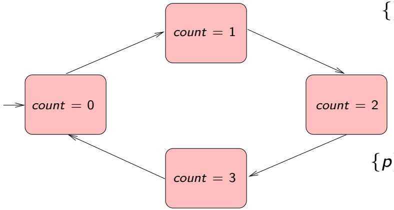
- Compile given property φ into an automaton $\mathcal{A}_{\neg\varphi}$ accepting precisely the models of $\neg\varphi$.
- Take the "product" of $\mathcal{T}$ and $\mathcal{A}_{\neg\varphi}$. (Pair states t of $\mathcal{T}$ and A of $\mathcal{A}_{\neg\varphi}$ together iff the set of propositions p true in t is exactly $A \cap P$.)
- Look for an "accepting" path in this product.
- If such a path exists, this is a counter-example to the claim that $\mathcal{T}$ satisfies the property φ.
- If no such path exists, then $\mathcal{T}$ satisfies φ.

Exercise

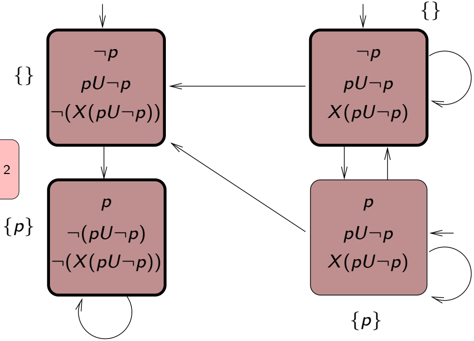
If p is the proposition "$count \neq 2$" then check if the mod-4 counter transition system satisfies the formula $\neg(pU\neg p)$.

- Construct the product of the mod-4 counter transition system and formula automaton for $pU\neg p$.
- Describe your counter-example if any.

Exercise



$count = 0$
$count = 1$
$count = 2$
$count = 3$



$\{\}$
$\neg p$
$pU\neg p$
$\neg(X(pU\neg p))$
$\{\}$
$\neg p$
$pU\neg p$
$X(pU\neg p)$
$\{p\}$
p
$\neg(pU\neg p)$
$\neg(X(pU\neg p))$
p
$pU\neg p$
$X(pU\neg p)$
$\{p\}$

Material for Temporal Logic based model-checking

- Textbook by Clarke, Grumberg, and Peled: *Model Checking*.
- Textbook by Christel Baier and Joost-Pieter Katoen: *Principles of Model Checking*, MIT Press 2008.