CS 512, Spring 2015, Handout 23

Model Checking: Algorithms for CTL and LTL

Assaf Kfoury

April 7, 2015
the CTL model-checking algorithm

Reading: [LCS, 3.6.1]

- the basic algorithm for CTL model-checking processes the given CTL WFF $\varphi$ (one of the two inputs) “from the inside out”, i.e., it starts from the smallest sub-WWF’s and works outwards towards $\varphi$.

- pseudo-code for the basic CTL model-checking algorithm is shown in [LCS, page 227].

- the state-explosion problem, [LCS, page 229]: e.g., adding a new propositional atom $p$ doubles the complexity of verifying a property involving $p$. 
an example using the CTL model-checking algorithm

INPUT:

1. \( \varphi \triangleq \textbf{AX EF } \psi \)
   where 
   \( \psi \triangleq (\neg p \land q \land \neg r) \lor (p \land \neg q \land r) \)

2. transition system \( \mathcal{M} \) (on the right, start states not specified)
an example using the CTL model-checking algorithm

INPUT:

1. \( \varphi \triangleq \text{AX EF } \psi \)

   where

   \( \psi \triangleq (\neg p \land q \land \neg r) \lor (p \land \neg q \land r) \)

2. transition system \( \mathcal{M} \) (on the right, start states not specified)

   blue states are where sub-WFF: 
   \( (\neg p \land q \land \neg r) \) is satisfied
an example using the CTL model-checking algorithm

INPUT:

1. $\varphi \triangleq \text{AX EF } \psi$
   where
   $\psi \triangleq (\neg p \land q \land \neg r) \lor (p \land \neg q \land r)$

2. transition system $\mathcal{M}$ (on the right, start states not specified)

   blue states are where sub-WFF: $(p \land \neg q \land r)$ is satisfied
an example using the CTL model-checking algorithm

INPUT:

1. $\varphi \triangleq \text{AX EF } \psi$
   where
   $\psi \triangleq (\neg p \land q \land \neg r) \lor (p \land \neg q \land r)$

2. transition system $\mathcal{M}$ (on the right, start states not specified)

   blue states are where sub-WFF:
   $\psi \triangleq (\neg p \land q \land \neg r) \lor (p \land \neg q \land r)$
   is satisfied
an example using the CTL model-checking algorithm

INPUT:

1. \( \varphi \triangleq \text{AX EF } \psi \)
   where
   \( \psi \triangleq (\neg p \land q \land \neg r) \lor (p \land \neg q \land r) \)

2. transition system \( \mathcal{M} \) (on the right, start states not specified)

   **red states** are where sub-WFF: (EF \( \psi \)) is satisfied
an example using the CTL model-checking algorithm

INPUT:

1. $\varphi \triangleq \text{AX EF } \psi$
   where
   $\psi \triangleq (\neg p \land q \land \neg r) \lor (p \land \neg q \land r)$

2. transition system $\mathcal{M}$ (on the right, start states not specified)

**double-red states** are where
sub-WFF: $\text{AX (EF } \psi \text{)}$ is satisfied

▶ **Conclusion:** $\mathcal{M} \models \varphi$ iff
$s_6$ is one of the start states of $\mathcal{M}$
more on the CTL model-checking algorithm

1. **Exercise**: Determine the states of $M$ (as defined in the preceding pages) satisfying the following CTL WFF’s:

   \[ \varphi_1 \triangleq \text{EG } q \]
   \[ \varphi_2 \triangleq \text{A } (p \text{ U } \varphi_1) = \text{A } (p \text{ U } (\text{EG } q)) \]
   \[ \varphi_3 \triangleq \text{EX } \varphi_2 = \text{EX} (\text{A } (p \text{ U } (\text{EG } q))) \]

2. **Problem** – illustrating abstraction (next slide) to alleviate *state-explosion*:
   For an arbitrary CTL WFF $\varphi$ and an arbitrary transition system $M$, let $M[\varphi]$ be the transition system obtained as follows:
   \[ \text{For every state } s, \text{ if } M, s \not\models \varphi, \text{ then delete state } s \text{ and all transitions (i.e., edges) to } s \text{ and all transitions from } s. \]

   Prove the following statement:
   \[ \text{For a state } s \text{ in transition system } M \text{ and CTL WFF } \varphi, \text{ we have that } M, s \models \text{EG } \varphi \text{ iff two conditions:} \]
   \[ \text{(i) } M, s \models \varphi \text{ and} \]
   \[ \text{(ii) there is a strongly-connected component of } M[\varphi] \text{ with at least one transition (i.e., edge) which is reachable from } s. \]
dealing with the state-explosion problem

Reading: [LCS, pp 229-230]. Different approaches:

- Efficient data structures: ordered binary decision diagrams (OBDD's), which represent sets of states instead of individual states, studied in [LCS, Chapter 6].
- Abstraction: More "abstract" models, i.e., building transition systems with fewer details or no details affecting satisfaction of the WFF to be checked.
- Partial order reduction: For asynchronous systems, several interleaving of component "traces" (see lecture notes of March 24, and again in this handout from slide 26 and on) may be equivalent for the satisfaction of the WFF to be checked.
- Induction: Model-checking systems with "large" numbers of identical, or similar, components can often be implemented by some sort of induction on this number.
- Composition: Break the verification problem down into several simpler verification problems... (more in lecture).
dealing with the state-explosion problem

Reading: [LCS, pp 229-230]. Different approaches:

▶ Efficient data structures: ordered binary decision diagrams (OBDD’s), which represent sets of states instead of individual states, studied in [LCS, Chapter 6].
Reading: [LCS, pp 229-230]. Different approaches:

- **Efficient data structures**: ordered binary decision diagrams (OBDD’s), which represent *sets* of states instead of individual states, studied in [LCS, Chapter 6].

- **Abstraction**: More “abstract” models, *i.e.*, building transition systems with fewer details or no details affecting satisfaction of the WFF to be checked.
dealing with the state-explosion problem

Reading: [LCS, pp 229-230]. Different approaches:

- **Efficient data structures**: ordered binary decision diagrams (OBDD’s), which represent sets of states instead of individual states, studied in [LCS, Chapter 6].

- **Abstraction**: More “abstract” models, i.e., building transition systems with fewer details or no details affecting satisfaction of the WFF to be checked.

- **Partial order reduction**: For asynchronous systems, several interleaving of component “traces” (see lecture notes of March 24, and again in this handout from slide 26 and on) may be equivalent for the satisfaction of the WFF to be checked.
dealing with the state-explosion problem

Reading: [LCS, pp 229-230]. Different approaches:

- **Efficient data structures**: *ordered binary decision diagrams* (OBDD’s), which represent *sets* of states instead of individual states, studied in [LCS, Chapter 6].

- **Abstraction**: More “abstract” models, *i.e.*, building transition systems with fewer details or no details affecting satisfaction of the WFF to be checked.

- **Partial order reduction**: For asynchronous systems, several interleaving of component “traces” (see lecture notes of March 24, and again in this handout from slide 26 and on) may be equivalent for the satisfaction of the WFF to be checked.

- **Induction**: Model-checking systems with “large” numbers of identical, or similar, components can often be implemented by some sort of induction on this number.
dealing with the state-explosion problem

Reading: [LCS, pp 229-230]. Different approaches:

- **Efficient data structures**: ordered binary decision diagrams (OBDD’s), which represent sets of states instead of individual states, studied in [LCS, Chapter 6].

- **Abstraction**: More “abstract” models, i.e., building transition systems with fewer details or no details affecting satisfaction of the WFF to be checked.

- **Partial order reduction**: For asynchronous systems, several interleaving of component “traces” (see lecture notes of March 24, and again in this handout from slide 26 and on) may be equivalent for the satisfaction of the WFF to be checked.

- **Induction**: Model-checking systems with “large” numbers of identical, or similar, components can often be implemented by some sort of induction on this number.

- **Composition**: Break the verification problem down into several simpler verification problems . . . (more in lecture).
the LTL model-checking algorithm

Reading: [LCS, 3.6.3, pp 232-238] – highly condensed, more difficult to read.
minimal (and different) presentation of LTL

\[
\phi, \psi ::= \top | p | \neg \phi | \phi \land \psi | X \phi | \phi U \psi
\]
minimal (and different) presentation of LTL

- syntax of LTL (compare with Handout 18, page 3):

\[ \varphi, \psi ::= T \mid p \mid \neg \varphi \mid \varphi \land \psi \mid X \varphi \mid \varphi U \psi \]
minimal (and different) presentation of LTL

- let $p$ range over a set $\text{AP}$ of atomic propositions $\text{AP} = \{p_1, \ldots, p_n\}$
minimal (and different) presentation of LTL

- let $p$ range over a set $AP$ of atomic propositions $AP = \{p_1, \ldots, p_n\}$

- let $2^{AP}$ be the set of all truth-value assignments to $AP$, i.e.,

$$2^{AP} = \left\{ \left(A(p_1), \ldots, A(p_n)\right) \mid A : AP \to \{\text{true, false}\} \right\}$$

Assaf Kfoury, CS 512, Spring 2015, Handout 23
let $p$ range over a set $\text{AP}$ of atomic propositions $\text{AP} = \{p_1, \ldots, p_n\}$

let $2^{\text{AP}}$ be the set of all truth-value assignments to $\text{AP}$, i.e.,

$$2^{\text{AP}} = \{ (A(p_1), \ldots, A(p_n)) \mid A : \text{AP} \rightarrow \{\text{true}, \text{false}\} \}$$

let $\sigma$ range over all $\omega$-sequences/$\omega$-words over $2^{\text{AP}}$, i.e., $\sigma \in (2^{\text{AP}})^\omega$
minimal (and different) presentation of LTL

- let $p$ range over a set $AP$ of atomic propositions $AP = \{p_1, \ldots, p_n\}$

- let $2^{AP}$ be the set of all truth-value assignments to $AP$, i.e.,

$$2^{AP} = \left\{ (A(p_1), \ldots, A(p_n)) \mid A : AP \to \{\text{true, false}\} \right\}$$

- let $\sigma$ range over all $\omega$-sequences/$\omega$-words over $2^{AP}$, i.e., $\sigma \in (2^{AP})^\omega$

Strictly speaking, to avoid technical problems later (e.g., see Remark at the end of page 31), we should restrict $\sigma$ to range over the “largest” (and countable) $\omega$-regular subset of the uncountable $(2^{AP})^\omega$. The largest $\omega$-regular language over a finite alphabet, e.g., $\{a, b\}$, can be defined by the $\omega$-regular expression $(a + b)^\omega$. 
minimal (and different) presentation of LTL

- Let $p$ range over a set $\text{AP}$ of atomic propositions $\text{AP} = \{p_1, \ldots, p_n\}$

- Let $2^{\text{AP}}$ be the set of all truth-value assignments to $\text{AP}$, i.e.,

$$2^{\text{AP}} = \left\{ \left( A(p_1), \ldots, A(p_n) \right) \mid A : \text{AP} \to \{\text{true, false}\} \right\}$$

- Let $\sigma$ range over all $\omega$-sequences/$\omega$-words over $2^{\text{AP}}$, i.e., $\sigma \in (2^{\text{AP}})^\omega$

Strictly speaking, to avoid technical problems later (e.g., see Remark at the end of page 31), we should restrict $\sigma$ to range over the “largest” (and countable) $\omega$-regular subset of the uncountable $(2^{\text{AP}})^\omega$. The largest $\omega$-regular language over a finite alphabet, e.g., $\{a, b\}$, can be defined by the $\omega$-regular expression $(a + b)^\omega$.

- **Notational convention:** if $\sigma = A_0 \ A_1 \ A_2 \ \cdots$, then for all $0 \leq i \leq j$,

$$\sigma[i] = A_i \quad \sigma[i..] = A_i \ A_{i+1} \ A_{i+2} \ \cdots \quad \sigma[i..j] = A_i \ A_{i+1} \ \cdots \ A_j$$
minimal (and different) presentation of LTL

- semantics of LTL (compare with Handout 18, page 4 and on) – NO mention of transition systems so far, the interpretation is with respect to an ω-word $\sigma \in (2^{AP})^\omega$: 

1. $\sigma \-verticalbar \equiv \top$
2. $\sigma \-verticalbar = p \iff p \in \sigma[0]$, i.e., $\sigma[0] \-verticalbar = p$
3. $\sigma \-verticalbar = \neg \phi \iff \sigma \-verticalbar \neq \phi$
4. $\sigma \-verticalbar = \phi \land \psi \iff \sigma \-verticalbar = \phi \\text{and} \ \sigma \-verticalbar = \psi$
5. $\sigma \-verticalbar = X \phi \iff \sigma[1...\] \-verticalbar = \phi$
6. $\sigma \-verticalbar = \phi \mathbf{U} \psi \iff \exists j \geq 0 \ \sigma[j...\] \-verticalbar = \psi \\text{and} \ \sigma[i...\] \-verticalbar = \phi \ \text{for every} \ 0 \leq i < j$
semantics of LTL (compare with Handout 18, page 4 and on) – NO mention of transition systems so far, the interpretation is with respect to an \( \omega \)-word \( \sigma \in (2^{AP})^\omega \):

1. \( \sigma \models T \)

2. \( \sigma \models p \) iff \( p \in \sigma[0] \), i.e., \( \sigma[0] \models p \)

3. \( \sigma \models \neg \varphi \) iff \( \sigma \not\models \varphi \)

4. \( \sigma \models \varphi \land \psi \) iff \( \sigma \models \varphi \) and \( \sigma \models \psi \)

5. \( \sigma \models X \varphi \) iff \( \sigma[1...] \models \varphi \)

6. \( \sigma \models \varphi \mathbf{U} \psi \) iff there is \( j \geq 0 \) such \( \sigma[j..] \models \psi \) and \( \sigma[i..] \models \varphi \) for every \( 0 \leq i < j \)
minimal (and different) presentation of LTL

- interpretation of a WFF $\varphi$ of LTL as a subset of $(2^{AP})^\omega$:

$$\omega\text{-words}(\varphi) \triangleq \left\{ \sigma \in (2^{AP})^\omega \mid \sigma \models \varphi \right\}$$
minimal (and different) presentation of LTL

- interpretation of a WFF $\varphi$ of LTL as a subset of $(2^{AP})^\omega$:

  $$\omega\text{-words}(\varphi) \triangleq \left\{ \sigma \in (2^{AP})^\omega \mid \sigma \models \varphi \right\}$$

- we mention transition systems for the first time now . . .

  let $M = (S, \to, L)$ be a transition system over AP
  let $\pi$ be an infinite execution path in $M$
  let $\text{trace}(\pi)$ be the $\omega$-word in $(2^{AP})^\omega$ induced by $\pi$
minimal (and different) presentation of LTL

- interpretation of a WFF $\varphi$ of LTL as a subset of $(2^{AP})^\omega$:

$$\omega\text{-words}(\varphi) \triangleq \left\{ \sigma \in (2^{AP})^\omega \mid \sigma \models \varphi \right\}$$

- we mention transition systems for the first time now . . .

  let $\mathcal{M} = (S, \rightarrow, L)$ be a transition system over AP
  let $\pi$ be an infinite execution path in $\mathcal{M}$
  let $\text{trace}(\pi)$ be the $\omega$-word in $(2^{AP})^\omega$ induced by $\pi$

- $\pi \models \varphi$ iff $\text{trace}(\pi) \models \varphi$
interpretation of a WFF $\varphi$ of LTL as a subset of $(2^{AP})^\omega$:

$$\omega\text{-words}(\varphi) \triangleq \left\{ \sigma \in (2^{AP})^\omega \mid \sigma \models \varphi \right\}$$

we mention transition systems for the first time now . . .

let $\mathcal{M} = (S, \rightarrow, L)$ be a transition system over AP

let $\pi$ be an infinite execution path in $\mathcal{M}$

let $\text{trace}(\pi)$ be the $\omega$-word in $(2^{AP})^\omega$ induced by $\pi$

$\pi \models \varphi$ iff $\text{trace}(\pi) \models \varphi$

$s \models \varphi$ iff $\pi \models \varphi$ for every execution path $\pi$ that starts at $s$
minimal (and different) presentation of LTL

▶ interpretation of a WFF $\varphi$ of LTL as a subset of $(2^{AP})^\omega$:

$$\omega\text{-words}(\varphi) \triangleq \left\{ \sigma \in (2^{AP})^\omega \mid \sigma \models \varphi \right\}$$

▶ we mention transition systems for the first time now . . .

let $\mathcal{M} = (S, \rightarrow, L)$ be a transition system over AP

let $\pi$ be an infinite execution path in $\mathcal{M}$

let $\text{trace}(\pi)$ be the $\omega$-word in $(2^{AP})^\omega$ induced by $\pi$

▶ $\pi \models \varphi$ iff $\text{trace}(\pi) \models \varphi$

▶ $s \models \varphi$ iff $\pi \models \varphi$ for every execution path $\pi$ that starts at $s$

▶ $\mathcal{M} \models \varphi$ iff $\text{Traces}(\mathcal{M}) \subseteq \omega\text{-words}(\varphi)$
model-checking algorithm for LTL

- basic idea is based on the following observation:

- for an arbitrary transition system $\mathcal{M}$ and an arbitrary WFF $\varphi$ of LTL

$$\mathcal{M} \models \varphi \quad \text{iff} \quad \text{Traces}(\mathcal{M}) \subseteq \omega\text{-words}(\varphi)$$

$$\text{iff} \quad \text{Traces}(\mathcal{M}) \cap \left((2^{AP})^\omega - \omega\text{-words}(\varphi)\right) = \emptyset$$

$$\text{iff} \quad \text{Traces}(\mathcal{M}) \cap \omega\text{-words}(\neg \varphi) = \emptyset$$
model-checking algorithm for LTL

- basic idea is based on the following observation:
- for an arbitrary transition system $\mathcal{M}$ and an arbitrary WFF $\varphi$ of LTL
  
  $\mathcal{M} \models \varphi$ iff \( \text{Traces}(\mathcal{M}) \subseteq \omega\text{-words}(\varphi) \)
  iff \( \text{Traces}(\mathcal{M}) \cap (\mathbb{2}^{2\text{AP}})^\omega - \omega\text{-words}(\varphi) = \emptyset \)
  iff \( \text{Traces}(\mathcal{M}) \cap \omega\text{-words}(\neg \varphi) = \emptyset \)

- hence, if $\mathcal{A}$ is a non-deterministic Buchi automaton such that
  
  \( \mathbb{L}_\omega(\mathcal{A}) = \omega\text{-words}(\neg \varphi) \)
  
  then $\mathcal{M} \models \varphi$ iff \( \text{Traces}(\mathcal{M}) \cap \mathbb{L}_\omega(\mathcal{A}) = \emptyset \)
model-checking algorithm for LTL

- basic idea is based on the following observation:
- for an arbitrary transition system $\mathcal{M}$ and an arbitrary WFF $\varphi$ of LTL

$$\mathcal{M} \models \varphi \quad \text{iff} \quad \text{Traces}(\mathcal{M}) \subseteq \omega\text{-words}(\varphi)$$
$$\quad \text{iff} \quad \text{Traces}(\mathcal{M}) \cap \left((2^{AP})^\omega - \omega\text{-words}(\varphi)\right) = \emptyset$$
$$\quad \text{iff} \quad \text{Traces}(\mathcal{M}) \cap \omega\text{-words}(\neg \varphi) = \emptyset$$

- hence, if $\mathcal{A}$ is a non-deterministic Buchi automaton such that

$$\mathcal{L}_\omega(\mathcal{A}) = \omega\text{-words}(\neg \varphi)$$

then $\mathcal{M} \models \varphi$ if and only if $
\text{Traces}(\mathcal{M}) \cap \mathcal{L}_\omega(\mathcal{A}) = \emptyset$

Remark: Strictly speaking, the equality $\mathcal{L}_\omega(\mathcal{A}) = \omega\text{-words}(\neg \varphi)$ cannot hold (why?). For it to hold, we need to restrict $\omega\text{-words}(\neg \varphi)$ to a $\omega$-regular subset . . . .
implementation of model-checking algorithm for LTL

two major steps:
implementation of model-checking algorithm for LTL

two major steps:

▶ for an arbitrary LTL WFF $\varphi$, construct an NBA $A$ representing $\neg \varphi$, i.e., construct $A$ over the alphabet $2^{AP}$ such that the $\omega$-regular language accepted by $A$ is precisely the set $\omega$-words($\neg \varphi$):

$$\mathcal{L}_\omega(A) = \omega\text{-words}(\neg \varphi)$$

http://www.lsv.ens-cachan.fr/~gastin/ltl2ba/index.php
implementation of model-checking algorithm for LTL

two major steps:

- for an arbitrary LTL WFF $\varphi$, construct an NBA $A$ representing $\neg \varphi$, i.e., construct $A$ over the alphabet $2^{AP}$ such that the $\omega$-regular language accepted by $A$ is precisely the set $\omega$-words($\neg \varphi$):

$$\mathcal{L}_\omega(A) = \omega\text{words}(\neg \varphi)$$

- construct the “product” NBA $B$ and determine whether $\mathcal{L}_\omega(B) = \emptyset$:

$$B \triangleq M \otimes A$$

http://www.lsv.ens-cachan.fr/~gastin/ltl2ba/index.php
two major steps:

- for an arbitrary LTL WFF $\varphi$, construct an NBA $A$ representing $\neg \varphi$, i.e., construct $A$ over the alphabet $2^{AP}$ such that the $\omega$-regular language accepted by $A$ is precisely the set $\omega$-words($\neg \varphi$):

$$\mathcal{L}_\omega(A) = \omega\text{-words}(\neg \varphi)$$

- construct the “product” NBA $B$ and determine whether $\mathcal{L}_\omega(B) = \emptyset$:

$$B \triangleq M \otimes A$$

- IF there is an $\omega$-word accepted by $B$,
  - THEN $\text{Traces}(M) \cap \mathcal{L}_\omega(A) \neq \emptyset$ and $M \not\models \varphi$,
  - ELSE $\text{Traces}(M) \cap \mathcal{L}_\omega(A) = \emptyset$ and $M \models \varphi$