Query Checking for Linear Temporal Logic

Samuel Huang

University of Maryland, Dept of CS

April 14th, 2017

Verification

Model checking: "Does model $\mathcal{M}$ satisfy property ϕ?"

Verification

Model checking: "Does model $\mathcal{M}$ satisfy property ϕ?"

Discovery

For model $\mathcal{M}$, what is the set of properties Φ that it satisfies?

Verification

Model checking: "Does model $\mathcal{M}$ satisfy property ϕ?"

Discovery

For model $\mathcal{M}$, what is the set of properties Φ that it satisfies?

Query Checking

For a model $\mathcal{M}$ and a property *template* $\phi[x]$, what is a solution c for x such that $\mathcal{M}$ satisfies $\phi[x := c]$?

Example

"What conditions must be met for the car to eventually stop?"

- Restrict our properties of interest to LTL formulas.

Outline

1. Linear Temporal Logic
2. LTL Formulas as Automata
3. LTL Model Checking
4. LTL Query Checking
5. Edge Shattering

Outline

1. Linear Temporal Logic
2. LTL Formulas as Automata
3. LTL Model Checking
4. LTL Query Checking
5. Edge Shattering

Some Related Work

- Query checking originally for CTL (Chan CAV 2000, Gurfinkel TSE 03)
- Earlier work in LTL query checking (Chockler HVC 2010)

Linear Temporal Logic (LTL)

Modal logic, useful for reasoning about outcomes of sequences or paths.

Linear Temporal Logic (LTL)

Modal logic, useful for reasoning about outcomes of sequences or paths.

- $\mathbf{X}$ *rain* – Tomorrow will be rainy.

Linear Temporal Logic (LTL)

Modal logic, useful for reasoning about outcomes of sequences or paths.

- $\mathbf{X}\, rain$ – Tomorrow will be rainy.
- $red\ \mathbf{U}\ green$ – The traffic light is red until it is green.

Linear Temporal Logic (LTL)

Modal logic, useful for reasoning about outcomes of sequences or paths.

- **X** *rain* – Tomorrow will be rainy.
- *red* **U** *green* – The traffic light is red until it is green.
- **G** *sunny* – It is always sunny (in Philadelphia).

Linear Temporal Logic (LTL)

Modal logic, useful for reasoning about outcomes of sequences or paths.

- $\mathbf{X}\, \mathit{rain}$ – Tomorrow will be rainy.
- $\mathit{red}\, \mathbf{U}\, \mathit{green}$ – The traffic light is red until it is green.
- $\mathbf{G}\, \mathit{sunny}$ – It is always sunny (in Philadelphia).
- $\mathbf{G}(\mathit{button} \rightarrow \mathbf{F}\, \mathit{stop})$ – Whenever the button is pressed, the machine eventually stops.

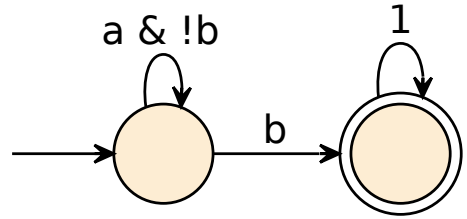
LTL formulas as automata

- Can think of LTL formulas as automata that accept (infinite) sequences of inputs:

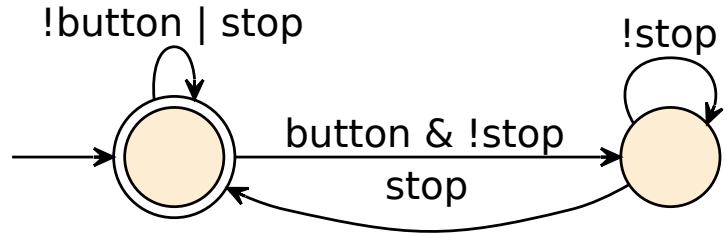
LTL formulas as automata

- Can think of LTL formulas as automata that accept (infinite) sequences of inputs:

$a \, \mathbf{U} \, b \quad \Longrightarrow$



$\mathbf{G}\,(button \rightarrow \mathbf{F}\, stop) \quad \Longrightarrow$

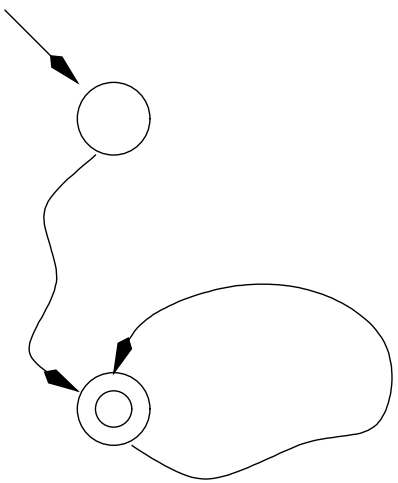


- As automata, facilitates LTL model checking when using similar style automata for data.
- Want to work in this representation to discover new properties.

LTL Model Checking

Automaton-based methods (Vardi, Wolper 86)

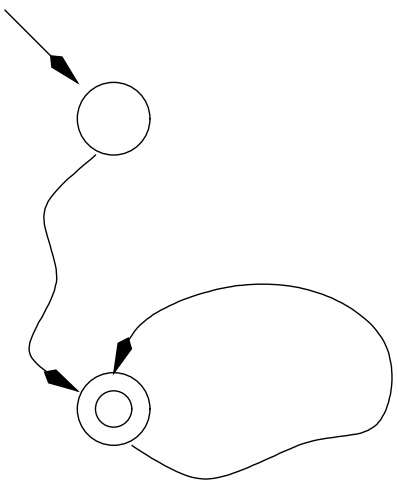
1. Take model $\mathcal{M}$, convert to an automaton representation B_M.
2. Take LTL formula ϕ, compute automaton for $\neg\phi$: $B_{\neg\phi}$.
3. Compute a composed automaton $B_c = B_M \cap B_{\neg\phi}$
4. Determine if $L(B_c) = \emptyset$



LTL Model Checking

Automaton-based methods (Vardi, Wolper 86)

1. Take model $\mathcal{M}$, convert to an automaton representation B_M.
2. Take LTL formula ϕ, compute automaton for $\neg\phi$: $B_{\neg\phi}$.
3. Compute a composed automaton $B_c = B_M \cap B_{\neg\phi}$
4. Determine if $L(B_c) = \emptyset$



LTL Templates from LTL Formulas

- An LTL template contains some unknown.
- Relax one part of an LTL formula to be variable (placeholder).

$$\begin{array}{lcl} \mathbf{G}\, \mathit{sunny} & \Longrightarrow & \mathbf{G}\ \ x \\ \mathbf{G}\,\mathbf{F}\, \mathit{halts} & \Longrightarrow & \mathbf{G}\,\mathbf{F}\, x \\ \mathbf{G}(a \rightarrow \mathbf{F}\, b) & \Longrightarrow & \mathbf{G}(x \rightarrow \mathbf{F}\, b) \end{array}$$

LTL Templates from LTL Formulas

- An LTL template contains some unknown.
- Relax one part of an LTL formula to be variable (placeholder).

$$\begin{array}{lcl} \mathbf{G}\, \mathit{sunny} & \Longrightarrow & \mathbf{G}\ \ x \\ \mathbf{G}\,\mathbf{F}\, \mathit{halts} & \Longrightarrow & \mathbf{G}\,\mathbf{F}\, x \\ \mathbf{G}(a \rightarrow \mathbf{F}\, b) & \Longrightarrow & \mathbf{G}(x \rightarrow \mathbf{F}\, b) \end{array}$$

- Can we modify existing LTL model checking approach to help us perform discovery?

Automaton Based Query Checking

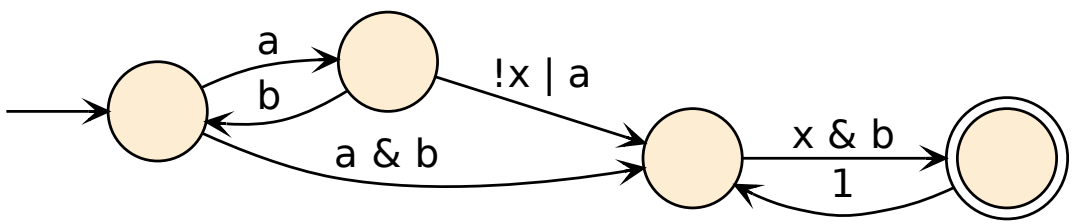
From LTL Model Checking:

- Compute a composed automaton $B_c = B_M \cap B_{\neg\phi}$
  - What do the transition labels look like on B_c?

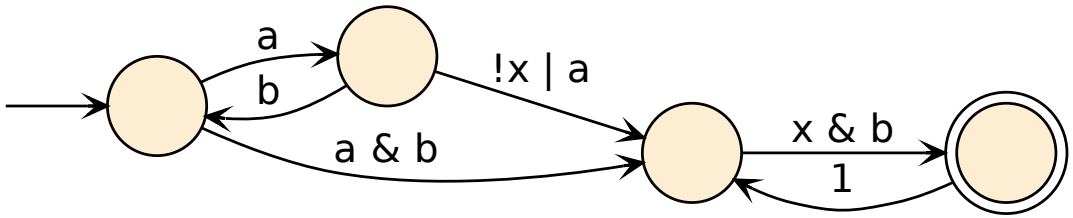
Automaton Based Query Checking

From LTL Model Checking:

- Compute a composed automaton $B_c = B_M \cap B_{\neg\phi}$
  - What do the transition labels look like on B_c?
- Determine if $L(B_c) = \emptyset$
  - How is this determined given the new edge labels?
  - The choice of grounding for the variable can invalidate edges

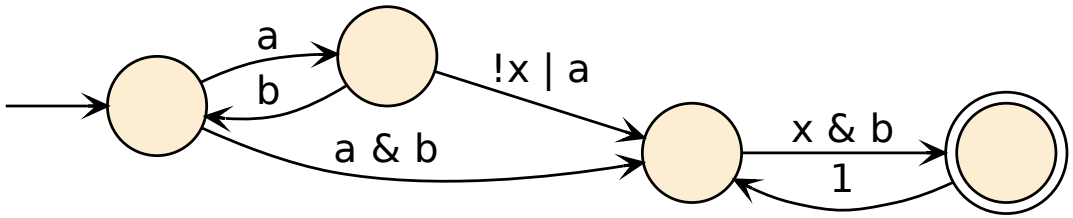


Büchi Propositional Automata



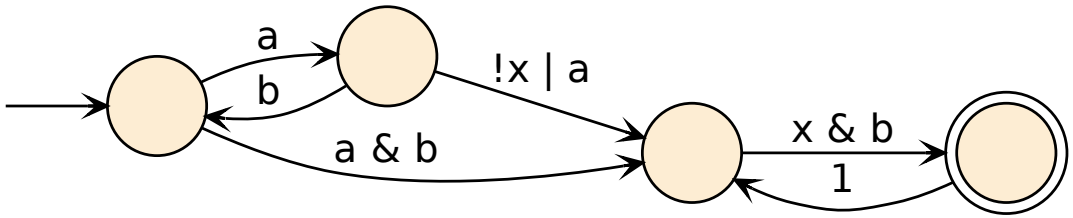
- Automaton representation of infinite length languages.

Büchi Propositional Automata



- Automaton representation of infinite length languages.
- Standard product composition of Büchi automata uses single alphabet symbols.

Büchi Propositional Automata



- Automaton representation of infinite length languages.
- Standard product composition of Büchi automata uses single alphabet symbols.
- Because an LTL template contains a variable, labels along transitions of the composition Büchi will contain propositional formulas as well.

Shattering

Sub-problem: Shattering an edge

Given a propositional formula with variable x, what assignments of x can make the formula to be logically equivalent to false?

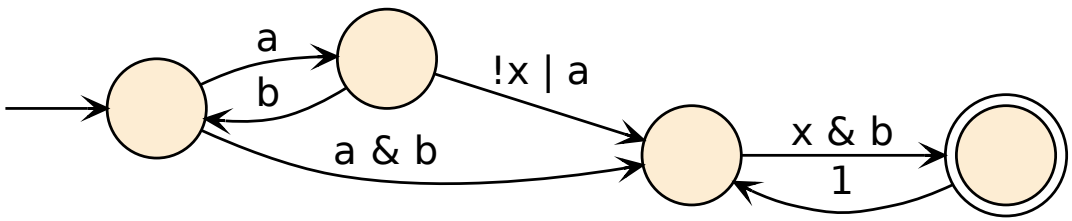
Shattering

Sub-problem: Shattering an edge

Given a propositional formula with variable x, what assignments of x can make the formula to be logically equivalent to false?

Shattering a Büchi automaton

Which edges can/must be shattered to make $L(B_c) = \emptyset$?



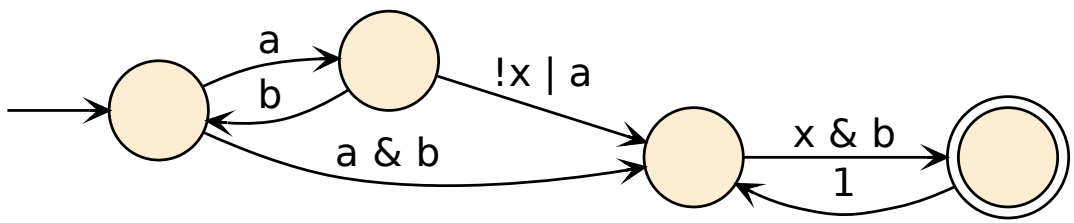
Shattering

Sub-problem: Shattering an edge

Given a propositional formula with variable x, what assignments of x can make the formula to be logically equivalent to false?

Shattering a Büchi automaton

Which edges can/must be shattered to make $L(B_c) = \emptyset$?



- Choice of edge set $\longrightarrow$ constraints on assignments for x

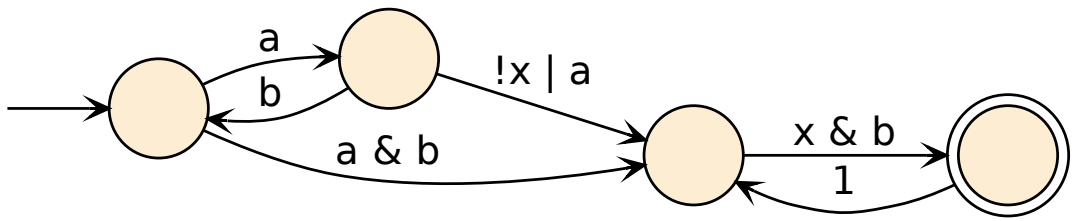
Shattering

Sub-problem: Shattering an edge

Given a propositional formula with variable x, what assignments of x can make the formula to be logically equivalent to false?

Shattering a Büchi automaton

Which edges can/must be shattered to make $L(B_c) = \emptyset$?



- Choice of edge set $\longrightarrow$ constraints on assignments for x
- Resolve all constraints to produce solutions for x.

Conclusion

Summary

- Developed automaton-based approach for LTL query checking
- "Shattering condition" problem yielded interesting side results

Ongoing & Future Work

- Expanding selection of datasets for evaluation
- Allowing for multiple variables/larger uncertainty

Thanks!

srhuang@cs.umd.edu