Closure Under Stuttering


Desired property of LTL formulas is closure under stuttering: interpretation of the formula remains the same under state sequences that differ only by repeated states [Abadi, Lamport'91].

- Guaranteed [Lamport'94] for a subset of LTL without the $\circ$ operator.

Examples:

- $\boxdot a$ is closed under stuttering
- $\circ a$ is not closed under stuttering

Legend:

- $\circ a$ is false
- $\bullet a$ is true

Notation: $<<F>> - F$ is closed under stuttering
Using LTL to Specify Production Cell System

- Case study initiated by Forchungszentrum Informatik (FZI)
- Aimed to show applicability of formal methods to real-world examples

Example property:

\[ \text{The magnet of the crane may be deactivated only when the magnet is above the feedbelt} \]

Resulting LTL formula:

\[ \square (activate \land \neg \diamond activate) \Rightarrow \diamond (head\_ver = DOWN) \]

Is this formula closed under stuttering?!!

Related Work

- Determining whether an arbitrary LTL formula is closed under stuttering is PSPACE-complete [Peled, Wilke, Wolper’96]
  - Tableau-based, $$$ approach
- A computationally-feasible algorithm for determining closure under stuttering for a subclass of formulas has been proposed [Holzmann, Kupferman’96] but not implemented in SPIN
  - Algorithm cannot be applied by hand
  - How useful in practice?

Our goal:

- Want to have syntactical restrictions on LTL (like “no next state”) that guarantee that the resulting formula is closed under stuttering
- Want the approach to apply to real-life problems
Edges

\[ (activate \land \Box \neg activate) \Rightarrow \Diamond (head\_ver = DOWN) \]

an edge (a change of value)

Formally, if \( A \) is an LTL formula, then

\[ \uparrow A = \neg A \land O A \quad \text{-- up or rising edge} \]
\[ \downarrow A = A \land O \neg A \quad \text{-- down or falling edge} \]
\[ \Uparrow A = \uparrow A \lor \downarrow A \quad \text{-- any edge} \]

Example: \( \uparrow \Box A \)

Edges = events
(Logical) edges = signal edges

Main Result

Observation:

stuttering does not add or delete edges (or change their relative order)

\[ \bigcirc \quad \bigcirc \quad \bigcirc \quad \bigcirc \]

Theorem:

\[ \ll A \gg \land \ll B \gg \Rightarrow \ll \Diamond (\neg A \land O A \land O B) \gg \]

Proof: in [Paun99]
Some Properties of Edges

- Edges are related:
  \[ \uparrow \neg A = \downarrow A \]
  \[ \downarrow \neg A = \uparrow A \]
  \[ \uparrow \neg A = \uparrow A \]

- Edges interact with each other:
  \[ \downarrow \downarrow A = \downarrow A \]
  \[ \uparrow \downarrow A = \uparrow \downarrow A \]

- Edges interact with boolean operators:
  \[ \uparrow (A \land B) = (\uparrow A \land B) \lor (B \land \uparrow A) \]

- Edges interact with temporal operators
  \[ \uparrow \circ A = \circ \uparrow A \]
  \[ \downarrow \Box A = false \]
  \[ \downarrow \Diamond A = \downarrow A \land \Box \neg A \]
  \[ \uparrow (A \cup B) = \neg (A \lor B) \land \circ (A \cup B) \]

Some Properties of Closure Under Stuttering

\( a \) is a variable or a constant \( \Rightarrow \langle\langle a\rangle\rangle \)

\( \langle\langle a\rangle\rangle = \langle\langle \neg a\rangle\rangle \)

\( \langle\langle a\rangle\rangle \land \langle\langle b\rangle\rangle \Rightarrow \langle\langle a \land b\rangle\rangle \)

\( \langle\langle a\rangle\rangle \Rightarrow \langle\langle \Box a\rangle\rangle \)

\( \langle\langle a\rangle\rangle \Rightarrow \langle\langle \Diamond a\rangle\rangle \)

\( \langle\langle a\rangle\rangle \land \langle\langle b\rangle\rangle \Rightarrow \langle\langle a \cup b\rangle\rangle \)

\( \langle\langle a\rangle\rangle \land \langle\langle b\rangle\rangle \Rightarrow \langle\langle a \ast b\rangle\rangle , \)

where \( \ast \in \{ \land, \lor, \Rightarrow, \Leftarrow, \Leftarrow \} \)

Formulas of the form \( \langle\langle a\rangle\rangle \Rightarrow f(\uparrow A) \): edges \( \uparrow \) and \( \downarrow \) can be used interchangeably.

193

194
Closure Under Stuttering Properties

Property 1 (Existence)

$$\langle\langle A \rangle\rangle \land \langle\langle B \rangle\rangle \land \langle\langle C \rangle\rangle \Rightarrow \langle\langle \diamondsuit (\uparrow A \land OB \land C) \rangle\rangle$$

with simplified versions:

$$\langle\langle A \rangle\rangle \land \langle\langle B \rangle\rangle \Rightarrow \langle\langle \diamondsuit (\uparrow A) \rangle\rangle$$

$$\langle\langle A \rangle\rangle \land \langle\langle B \rangle\rangle \Rightarrow \langle\langle \diamondsuit (\uparrow A \land OB) \rangle\rangle$$

Property 2 (Universality)

$$\langle\langle A \rangle\rangle \land \langle\langle B \rangle\rangle \land \langle\langle C \rangle\rangle \Rightarrow \langle\langle \Box (\uparrow A \Rightarrow (OB \lor C)) \rangle\rangle$$

with simplified versions:

$$\langle\langle A \rangle\rangle \land \langle\langle B \rangle\rangle \Rightarrow \langle\langle \Box (\uparrow A \Rightarrow B) \rangle\rangle$$

$$\langle\langle A \rangle\rangle \land \langle\langle B \rangle\rangle \Rightarrow \langle\langle \Box (\uparrow A \Rightarrow OB) \rangle\rangle$$

Closure Under Stuttering Properties

(Cont’d)

Property 3 (Until)

$$\langle\langle A \rangle\rangle \land \langle\langle B \rangle\rangle \land \langle\langle C \rangle\rangle \land \langle\langle D \rangle\rangle \land \langle\langle E \rangle\rangle \land \langle\langle F \rangle\rangle$$

$$\Rightarrow \langle\langle (\lnot \uparrow A \lor OB \lor C) \lor (\uparrow D \land OB \land E \land F) \rangle\rangle$$

with many simplified versions.

Examples:

The magnet of the crane may be deactivated only when the magnet is above the feedbelt:

$$\Box(\Diamond activate \Rightarrow \Diamond (head\_ver = DOWN))$$

Initially, no items should be dropped on the table before the operator pushes and releases the GO button

$$\neg \downarrow hold \lor \downarrow button$$
Quick Summary

- We introduced the notion of edges for LTL
- We provided a set of theorems that enable syntax-based analysis of a large class of formulas for closure under stuttering.
- Such theorems can be added to a theorem-prover for mechanized checking.
  
  !! But the language of edges is not closed !!

Example: $\mathcal{A}$

Are the properties that can be identified using our method useful in practice?

Application: Property Patterns

- Pattern-based approach [Dwyer, Avrunin, Corbett'98,'99]
  - Presentation, codification and reuse of property specifications
  - Easy conversion between formalisms: CTL, LTL, QRE, GIL...
  - Goal: to enable novice users to express complex properties effectively
    - LTL properties are state-based
- Apply our theory to
  - extend the pattern-system with events for LTL properties
  - check closure-under-stuttering of resulting formulas
Pattern Hierarchy

- **Absence**: A condition does not occur within a scope.
- **Existence**: A condition must occur within a scope.
- **Universality**: A condition occurs throughout a scope.
- **Response**: A condition must always be followed by another within a scope.
- **Precedence**: A condition must always be preceded by another within a scope.

Scopes

Scopes are regions of interest over which the condition is evaluated.

- **Global**
- **Before R**
- **After Q**
- **Between Q and R**
- **After Q Until R**
- **State/Event Sequence**
Example

LTL formulation of the property

$S$ precedes $P$ between $Q$ and $R$

(Precedence pattern with "between $Q$ and $R$" scope) is

$\Box((Q \land \Diamond R) \Rightarrow \neg P \lor (S \lor R))$)

Note that $S, P, Q, R$ are states.

Extending the Pattern System

- Want to extend LTL patterns to reasoning about events
- "next" operator: are resulting properties closed under stuttering?

Assumptions:
- Multiple events can happen simultaneously
- Intervals are closed-left, open-right, as in original system

\[ Q \rightarrow R \]
Extending the Pattern System

- We have considered the following possibilities:
  1. \( P, S \) -- states \( Q, R \) -- states
  2. \( P, S \) -- states \( Q, R \) -- up edges
  3. \( P, S \) -- up edges \( Q, R \) -- up edges

Note: down edges can be substituted for up edges

- We extended Absence, Existence, Universality, Precedence, and Response patterns.

- Some of properties from other patterns, e.g. Chain Precedence, are not closed under stuttering [paun, chechik99]

A Note on Edges

Definition of an edge:
\[
\uparrow A = \neg A \land \diamond A
\]
Thus, an edge is detected in a state before it occurs.

Example: \( P \) always becomes true after \( Q \).

Formulations:
- \( \Box (Q \Rightarrow \Box P) \) if \( Q \) and \( P \) are states
- \( \Box (\uparrow Q \Rightarrow \Diamond \Box P) \) if \( P \) is a state and \( Q \) is an event
Extension of Patterns - Existence Pattern

- **P Exists Before R**
  0. ∆R \(\Rightarrow\) \(\neg(-P \land R)\)
  1. ∆\(\uparrow\)R \(\Rightarrow\) \((\neg\uparrow R \land P)\)
  2. ∆R \(\Rightarrow\) \(\neg(-\uparrow P \land R)\)
  3. ∆\(\uparrow\)R \(\Rightarrow\) \(\neg(-\uparrow P \land \uparrow R)\)

- **P Exists Between Q and R**
  0. □(Q \(\land\) ∆R \(\Rightarrow\) \(\neg(-P \land R) \land \neg R)\)
  1. □(\(\uparrow\)Q \(\land\) ∆\(\uparrow\)R \(\Rightarrow\) O(\(\neg\uparrow R \land P) \land \neg R)\)
  2. □(Q \(\land\) ∆R \(\Rightarrow\) \(\neg(-\uparrow P \land R) \land \neg R)\)
  3. □(\(\uparrow\)Q \(\land\) ∆\(\uparrow\)R \(\Rightarrow\) \(\neg(-\uparrow P \land \uparrow R) \land \neg \uparrow R)\)

Using the Pattern System: Example

Example property:

*The robot must weigh the blank after pickup from the feedbelt, but before depositing it on the press.*

Variables:

- **(state) mgn** - true when the magnet is on
- **(state) scl** - the scale reports a successful weighing

This is the **Existence** pattern: weighing (state) must happen between (events) pickup and deposit. Scope is **Between R and Q**.

Pattern Formula:

\[ \square(\uparrow Q \land \diamond \uparrow R \Rightarrow O(\neg \uparrow R \land P) \land \neg \uparrow R) \]

Resulting Formula:

\[ \square(\uparrow mgn \land \diamond \downarrow mgn \Rightarrow O(\neg \downarrow mgn \land scl) \land \neg \downarrow mgn) \]
Proving Closure Under Stuttering

- Can use properties of closure under stuttering, the algebra of edges, and rules of logic to show
  \[(<<P>> \land <<Q>> \land <<R>>) \Rightarrow \]
  \[<<\Box (\uparrow Q \land \uparrow R) \Rightarrow \Diamond (\neg \uparrow R \cup P) \land \neg \uparrow R)>>\]
  in roughly 8 steps (see paper) completely syntactically.

- We proved all new edge-based formulas for closure under stuttering.
- Users can use these without worrying

---

Summary of the Problem

- The “next” operator in LTL is required for reasoning about events
- “next” is present => the result is not closed under stuttering
- Solution: introduce extra variables to simulate events:
  - Clutter the model, make harder to analyze
  - Results of verification cannot be interpreted correctly, without complete understanding of the modeling language and LTL. So, novice users will be making mistakes!!!
Summary of Solution

- We introduced the notion of edges for LTL
- We provided a set of theorems that enable syntax-based analysis of a large class of formulas for closure under stuttering.
- Such theorems can be added to a theorem-prover for mechanized checking.
- The language is not closed (unlike “next”-free LTL)
- But it can express properties useful in practice:
  - Properties of Production Cell [Paun, Chechik, Biechele'98]
  - Property patterns + events [Paun, Chechik'99]
- For more information:
  http://www.cs.toronto.edu/~chechik/edges.html