On-The-Fly Partial Order Reduction

Lecture #11 of Advanced Model Checking

Joost-Pieter Katoen
Lehrstuhl 2: Software Modeling & Verification

E-mail: katoen@cs.rwth-aachen.de

May, 2014
Outline of partial-order reduction

• During state space generation obtain $\hat{TS}$
  – a reduced version of transition system $TS$ such that $\hat{TS} \triangleq TS$
  ⇒ this preserves all stutter sensitive LT properties, such as LTL $\Box$
  – at state $s$ select a (small) subset of enabled actions in $s$
  – different approaches on how to select such set: consider Peled’s ample sets

• Static partial-order reduction
  – obtain a high-level description of $\hat{TS}$ (without generating $TS$)
  ⇒ POR is preprocessing phase of model checking

• Dynamic (or: on-the-fly) partial-order reduction
  – construct $\hat{TS}$ during LTL $\Box$ model checking
  – if accept cycle is found, there is no need to generate entire $\hat{TS}$
Ample-set conditions for LTL

(A1) **Nonemptiness condition**
\[ \emptyset \neq \text{ample}(s) \subseteq \text{Act}(s) \]

(A2) **Dependency condition**
Let \( s \xrightarrow{\beta_1} \ldots \xrightarrow{\beta_n} s_n \xrightarrow{\alpha} t \) be a finite execution fragment in \( TS \) such that \( \alpha \) depends on \( \text{ample}(s) \). Then: \( \beta_i \in \text{ample}(s) \) for some \( 0 < i \leq n \).

(A3) **Stutter condition**
If \( \text{ample}(s) \neq \text{Act}(s) \) then any \( \alpha \in \text{ample}(s) \) is a stutter action.

(A4) **Cycle condition**
For any cycle \( s_0 s_1 \ldots s_n \) in \( \hat{TS} \) and \( \alpha \in \text{Act}(s_i) \), for any \( 0 < i \leq n \), there exists \( j \in \{ 1, \ldots, n \} \) such that \( \alpha \in \text{ample}(s_j) \).
Correctness theorem

For action-deterministic, finite $TS$ without terminal states:
if conditions (A1) through (A4) are satisfied, then $\widehat{TS} \triangleq TS$. 
Strong cycle condition

(A4’) Strong cycle condition
On any cycle $s_0 s_1 \ldots s_n$ in $\widehat{TS}$, there exists $j \in \{1, \ldots, n\}$ such that $ample(s_j) = Act(s_j)$.

- If (A1) through (A3) hold: (A4’) implies the cycle condition (A4)
- (A4’) can be checked easily in DFS when backward edge is found
The branching-time ample approach

- **Linear-time ample approach:**
  - during state space generation obtain \( \widehat{T}S \) such that \( \widehat{T}S \not\equiv TS \)
  - this preserves all stutter sensitive LT properties, such as LTL\( \setminus \bigcirc \)
  - static partial order reduction: generate \( \widehat{T}S \) prior to verification
  - on-the-fly partial order reduction: generate \( \widehat{T}S \) during the verification
  - generation of \( \widehat{T}S \) by means of static analysis of program graphs

- **Branching-time ample approach**
  - during state space generation obtain \( \widehat{T}S \) such that \( \widehat{T}S \approx^{\text{div}} TS \)
  - this preserves all CTL\( \setminus \bigcirc \) and CTL\( ^*\setminus \bigcirc \) formulas
  - static partial order reduction only

\[ as \approx^{\text{div}} \text{ is strictly finer than } \not\equiv, \text{ try (A1) through (A4)} \]
Example

transition system $TS$
Conditions (A1)-(A4) are insufficient

\[ \hat{TS} \models \forall \Box \left( a \rightarrow (\forall \Diamond b \lor \forall \Diamond c) \right) \] but \( TS \) does not and thus \( \hat{TS} \not\sim^{\text{div}} TS \)
Branching condition

(A5)
If $ample(s) \neq Act(s)$ then $|ample(s)| = 1$
A sound reduction for CTL* 

\[ \hat{TS} \not\models \forall \Box \left( a \rightarrow (\forall \Box b \lor \forall \Box c) \right) \quad \text{and} \quad TS \text{ does not } \approx^{\text{div}} TS \]
Correctness theorem

For action-deterministic, finite $TS$ without terminal states:
if conditions (A1) through (A5) are satisfied, then $\hat{TS} \approx^{\text{div}} TS$.

recall that this implies that $\hat{TS}$ and $TS$ are $\text{CTL}^{\lor -}$-equivalent
Ample-set conditions for CTL

(A1) **Nonemptiness condition**
\[ \emptyset \neq \text{ample}(s) \subseteq \text{Act}(s) \]

(A2) **Dependency condition**
Let \( s \xrightarrow{\beta_1} \ldots \xrightarrow{\beta_n} s_n \xrightarrow{\alpha} t \) be a finite execution fragment in \( TS \) such that \( \alpha \) depends on \( \text{ample}(s) \). Then: \( \beta_i \in \text{ample}(s) \) for some \( 0 < i \leq n \).

(A3) **Stutter condition**
If \( \text{ample}(s) \neq \text{Act}(s) \) then any \( \alpha \in \text{ample}(s) \) is a stutter action.

(A4) **Cycle condition**
For any cycle \( s_0 s_1 \ldots s_n \) in \( \widehat{TS} \) and \( \alpha \in \text{Act}(s_i) \), for some \( 0 < i \leq n \), there exists \( j \in \{ 1, \ldots, n \} \) such that \( \alpha \in \text{ample}(s_j) \).

(A5) **Branching condition**
If \( \text{ample}(s) \neq \text{Act}(s) \) then \( |\text{ample}(s)| = 1 \)

© JPK