State Space Reductions

Pavel Moravec

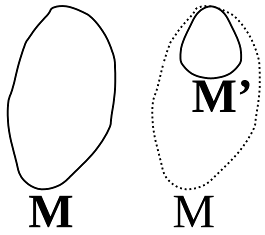
State Space Reductions project group
Parallel and Distributed Systems Laboratory

Seminar of ParaDiSe
spring 2006

Contents

- efficient checking of p.o.r. properties for LTL model checking
- LTL model checking based on regular expressions
- distributed algorithms for SCCs decomposition

Partial Order Reduction for LTL



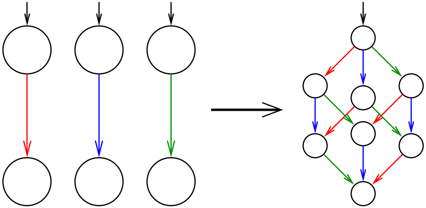
- $\varphi \in$ LTL-X: $M \models \varphi \Leftrightarrow M' \models \varphi$
- construction of M':
  - start from the initial state
  - for each reached state, explore a subset of enabled transitions

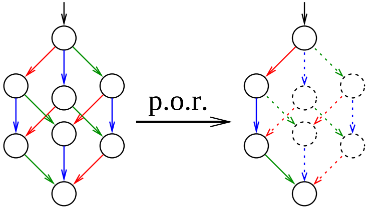
Partial Order Reduction for LTL

Conditions imposed on ample sets:

- C0: $\mathit{ample}(s) = \emptyset \Rightarrow \mathit{enabled}(s) = \emptyset$
- C1: every path in the full state space M starting at *s* can not have transition dependent on some transition in $\mathit{ample}(s)$ without a transition in $\mathit{ample}(s)$ occurring first
- C2: $\mathit{ample}(s)$ has visible transition $\Rightarrow \mathit{ample}(s) = \mathit{enabled}(s)$
- C3: for each cycle in M' holds: it contains state where α is enabled $\Rightarrow$ α is included in ample set of some state in cycle

Partial Order Reduction for LTL





Partial Order Reduction for LTL – Improvement

- in general approach, only transitions of a single process are taken as a candidate to an ample set
- if more dependencies among transitions occur, p.o.r. achieves small reduction
- an approach assuming ample sets contained transitions from different processes proposed

LTL Based on Regular Expressions

My goal: to propose an approach to LTL model checking which:

- doesn’t rely on construction of Büchi automaton
- isn’t forced to store all states
- is exploration independent

LTL Based on Regular Expressions

FOMLO = First Order Monadic Logic.

- formulae express properties of infinite words over finite alphabet
- example of a FOMLO formula:
  $f = \forall x_1 : \exists x_2 . x_2 > x_1 : x_2 \in P_a$
- LTL and FOMLO are expressively equivalent (*GFa* is equivalent to f)

LTL Based on Regular Expressions

SFRE = Star-free Regular Expressions.

- SFRE expressions are built from letters of an alphabet, $\emptyset$ and operands $\cdot$, $+$ and $\neg$
- Thomas (1979) has proven that FOMLO (LTL) and SFRE are expressively equivalent:
  - thus, for each FOMLO f there is an equivalent expression of the form $\bigcup U_i \cdot V_i^\omega$, where U_i, V_i are SFRE
  - equivalently, for each LTL formula φ, there is an equivalent expression of the form $E_\varphi = \bigcup U_i \cdot V_i^\omega$

LTL Based on Regular Expressions

Proposed approach: for a given model M and given LTL formula φ:

- find expression $E_{\neg\varphi}$
- test whether a behaviour of M satisfies $E_{\neg\varphi}$:
  - if so, $M \not\models \varphi$ and such a behaviour is a counterexample
  - if not so, $M \models \varphi$

Distributed SCCs Decomposition

Usage in model checking:

- LTL (property specified by GBA, Müler, or Street automaton)
- τ-confluence reduction
- probabilistic model checking (???)

Distributed SCCs Decomposition

Existing algorithms:

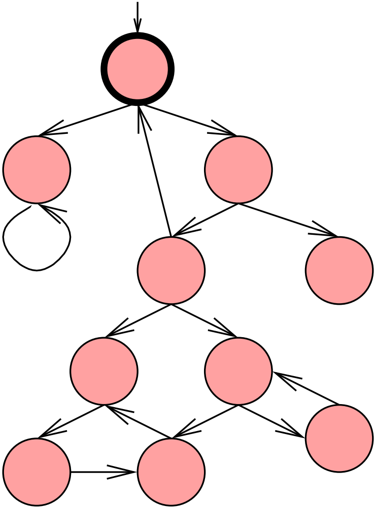
- sequential (Tarjan) – efficient distribution unknown
- FWD+BWD traversal
- colouring & colour prioritising (S. Orzan)

Distributed SCCs Decomposition

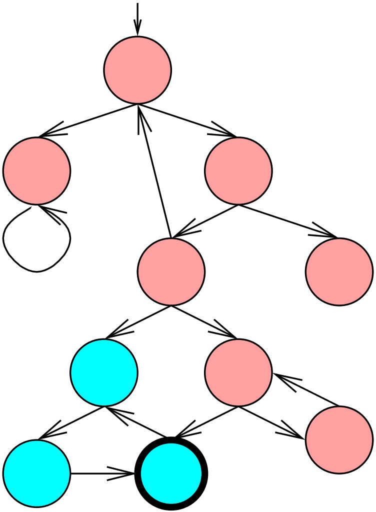
Our (my + Jiřík's) proposals:

- based on colouring a graph from all vertices in sequence
- storing additional information to reduce the number of reachabilities to be performed
- usage of back-level edges/states
- variants with back-propagations

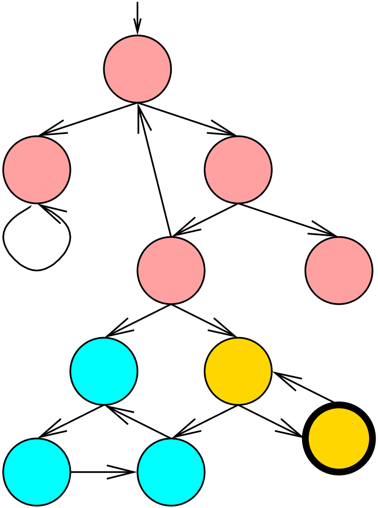
Algorithm Essence



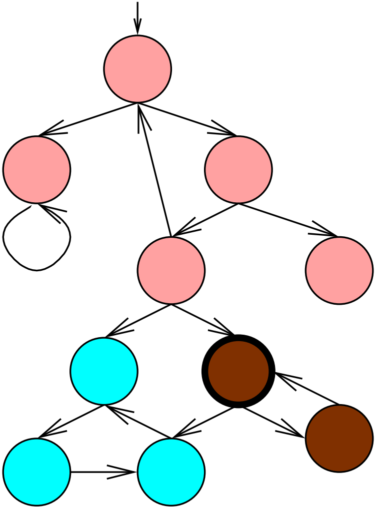
Algorithm Essence



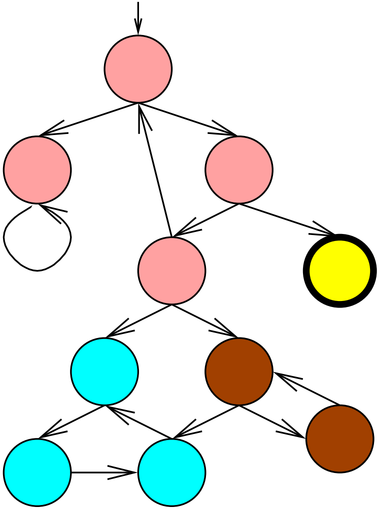
Algorithm Essence



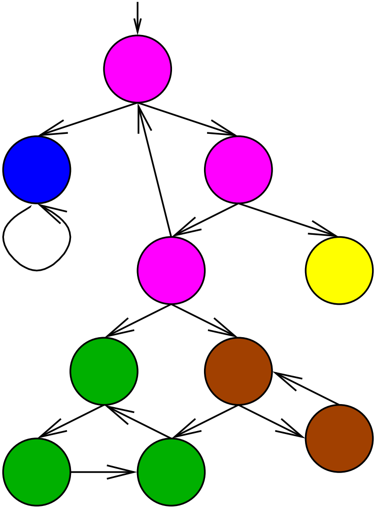
Algorithm Essence



Algorithm Essence



Algorithm Essence

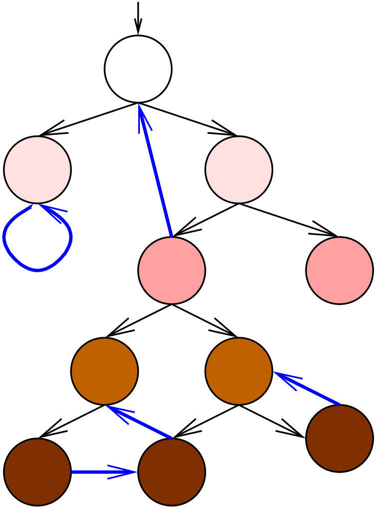


Distributed SCCs Decomposition

Our (Me + Jiřík) proposals:

- based on colouring a graph from all vertices in sequence
- storing additional information to reduce the number of reachabilities to be performed
- usage of back-level edges/states
- variants with back-propagations

Back Level Edges



Distributed SCCs Decomposition

Our (Me + Jiřík) proposals:

- based on colouring a graph from all vertices in sequence
- storing additional information to reduce the number of reachabilities to be performed
- usage of back-level edges/states
- variants with back-propagations

Distributed SCCs Decomposition

Results:

- algorithms based on forward traversal only: generally no great success
- improvement of FWD+BWD algorithm: significant speedup
- research still in progress