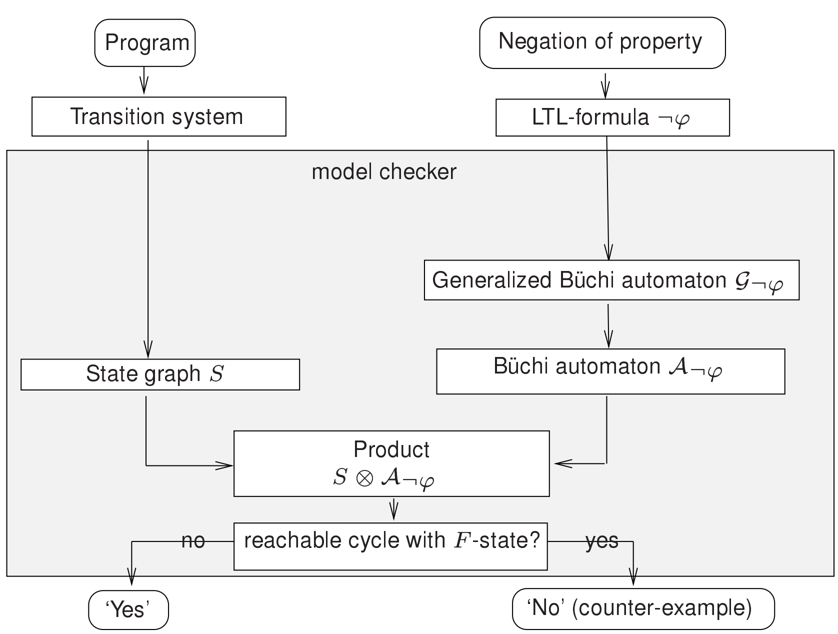
Verification – Lecture 12
Model Checking (Complexity)

Bernd Finkbeiner – Sven Schewe
Rayna Dimitrova – Lars Kuhtz – Anne Proetzsch

Wintersemester 2007/2008

LTL model checking



The LTL model checking problem

- Input:
  - LTL formula φ over atomic propositions *AP*
  - Finite state graph (Kripke structure) $S = (Q, Q_0, E, L)$ with
    - finite set of states Q,
    - initial states $Q_0 \subseteq Q$,
    - edges $E \subseteq Q \times Q$,
    - labeling function $L : Q \to 2^{AP}$.
- Output:
  - path π in T such that $\pi \models \neg\varphi$, or
  - "yes" if no such path exists.

From LTL to GNBA

GNBA $\mathcal{G}_\varphi$ over 2^{AP} for LTL-formula φ with $\mathcal{L}_\omega(\mathcal{G}_\varphi) = Words(\varphi)$:

- Assume φ only contains the operators $\wedge$, $\neg$, $\bigcirc$ and $\mathcal{U}$
  - $\vee$, $\rightarrow$, $\Diamond$, $\Box$, $\mathcal{W}$, and so on, are expressed in terms of these basic operators
- States are *elementary sets* of sub-formulas in φ
  - for $\sigma = A_0 A_1 A_2 \ldots \in Words(\varphi)$, expand $A_i \subseteq AP$ with sub-formulas of φ
  - ... to obtain the infinite word $\bar{\sigma} = B_0 B_1 B_2 \ldots$ such that

$$\psi \in B_i \qquad \text{if and only if} \qquad \sigma^i = A_i A_{i+1} A_{i+2} \ldots \models \psi$$

  - $\bar{\sigma}$ is intended to be a run in GNBA $\mathcal{G}_\varphi$ for σ
- Transitions are derived from semantics $\bigcirc$ and expansion law for $\mathcal{U}$
- Accept sets guarantee that: $\bar{\sigma}$ is an accepting run for σ iff $\sigma \models \varphi$

Complexity for LTL to NBA

For any LTL-formula φ (over *AP*) there exists an NBA $\mathcal{A}_\varphi$ with *Words*$(\varphi) = \mathcal{L}_\omega(\mathcal{A}_\varphi)$ and which can be constructed in time and space in $2^{\mathcal{O}(|\varphi|)}$.

Complexity

- States GNBA $\mathcal{G}_\varphi$ are elementary sets of formulae in *closure*(φ)
  - sets B can be represented by bit vectors with single bit per subformula ψ of φ
- The number of states in $\mathcal{G}_\varphi$ is bounded by $2^{|\text{subf}(\varphi)|}$
  - where subf(φ) denotes the set of all subformulae of φ
- The number of accepting sets of $\mathcal{G}_\varphi$ is bounded above by $\mathcal{O}(|\varphi|)$
- The number of states in NBA $\mathcal{A}_\varphi$ is thus bounded by $2^{\mathcal{O}(|\varphi)|} \cdot \mathcal{O}(|\varphi|)$
- $2^{\mathcal{O}(|\varphi|)} \cdot \mathcal{O}(|\varphi|) = 2^{\mathcal{O}(|\varphi|)}$ ■

Lower bound

There exists a family of LTL formulas φ_n with $|\varphi_n| = \mathcal{O}(poly(n))$

such that every NBA $\mathcal{A}_{\varphi_n}$ for φ_n has at least 2^n states

Product automaton

- Given:
  - state graph $S = (Q, Q_0, E, L)$,
  - Büchi automaton $\mathcal{A}_{\neg\varphi} = (Q', Q'_0, \delta', F')$
- Compute: Büchi automaton $S \otimes \mathcal{A}_{\neg\varphi} = (Q'', Q''_0, \delta'', F'')$
  - $Q'' = Q \times Q'$,
  - $Q''_0 = Q_0 \times Q'_0$,
  - $(q, q') \in \delta''((p, p'), A)$
    iff $A = L(p), q' \in \delta'(p', A)$, and $(p, q) \in E$.
  - $F'' = Q \times F'$

Nested depth-first search

- Idea: perform the two depth-first searches in an *interleaved* way
  - the outer DFS serves to encounter all reachable F-states
  - the inner DFS seeks for backward edges
- *Nested DFS*
  - on full expansion of F-state s in the outer DFS, start inner DFS
  - in inner DFS, visit all states reachable from s *not visited* in the inner DFS yet
  - no backward edge found? continue the outer DFS (look for next F state)
- *Counterexample generation*: DFS stack concatenation
  - stack U for the outer DFS = path fragment from $s_0 \in I$ to s (in reversed order)
  - stack V for the inner DFS = a cycle from state s to s (in reversed order)

Time complexity

The worst-case time complexity of nested DFS is in

$$\mathcal{O}(N+M)$$

where N is $\#$ reachable states in S, and M is $\#$ edges in state graph

Complexity for LTL model checking

The time and space complexity of LTL model checking is in $\mathcal{O}\left((M+N)\cdot 2^{|\varphi|}\right)$

On-the-fly LTL model checking

- Idea: find a counter-example *during* the generation of *Reach*(S) and $\mathcal{A}_{\neg\varphi}$
  - exploit the fact that *Reach*(S) and $\mathcal{A}_{\neg\varphi}$ can be generated in parallel

$\Rightarrow$ Generate *Reach*$(S \otimes \mathcal{A}_{\neg\varphi})$ "on demand"

  - consider a new vertex only if no accepting cycle has been found yet
  - only consider the successors of a state in $\mathcal{A}_{\neg\varphi}$ that match current state in S

$\Rightarrow$ Possible to find an accepting cycle without generating $\mathcal{A}_{\neg\varphi}$ entirely

The LTL model-checking problem is PSPACE-complete

The LTL model checking problem is PSPACE-complete.