Liveness Checking as Safety Checking to Find Shortest Counterexamples to Linear Time Properties

Viktor Schuppan

Computer Systems Institute, ETH Zürich

`http://www.inf.ethz.ch/~schuppan/`

Defense Thesis ETH 16268

September 28, 2005, Zürich, Switzerland

Safety vs. Liveness

[Lamport ’77], [Alpern, Schneider ’85]

Safety

“Something bad will not happen.”

The “bad thing” is irremediable.

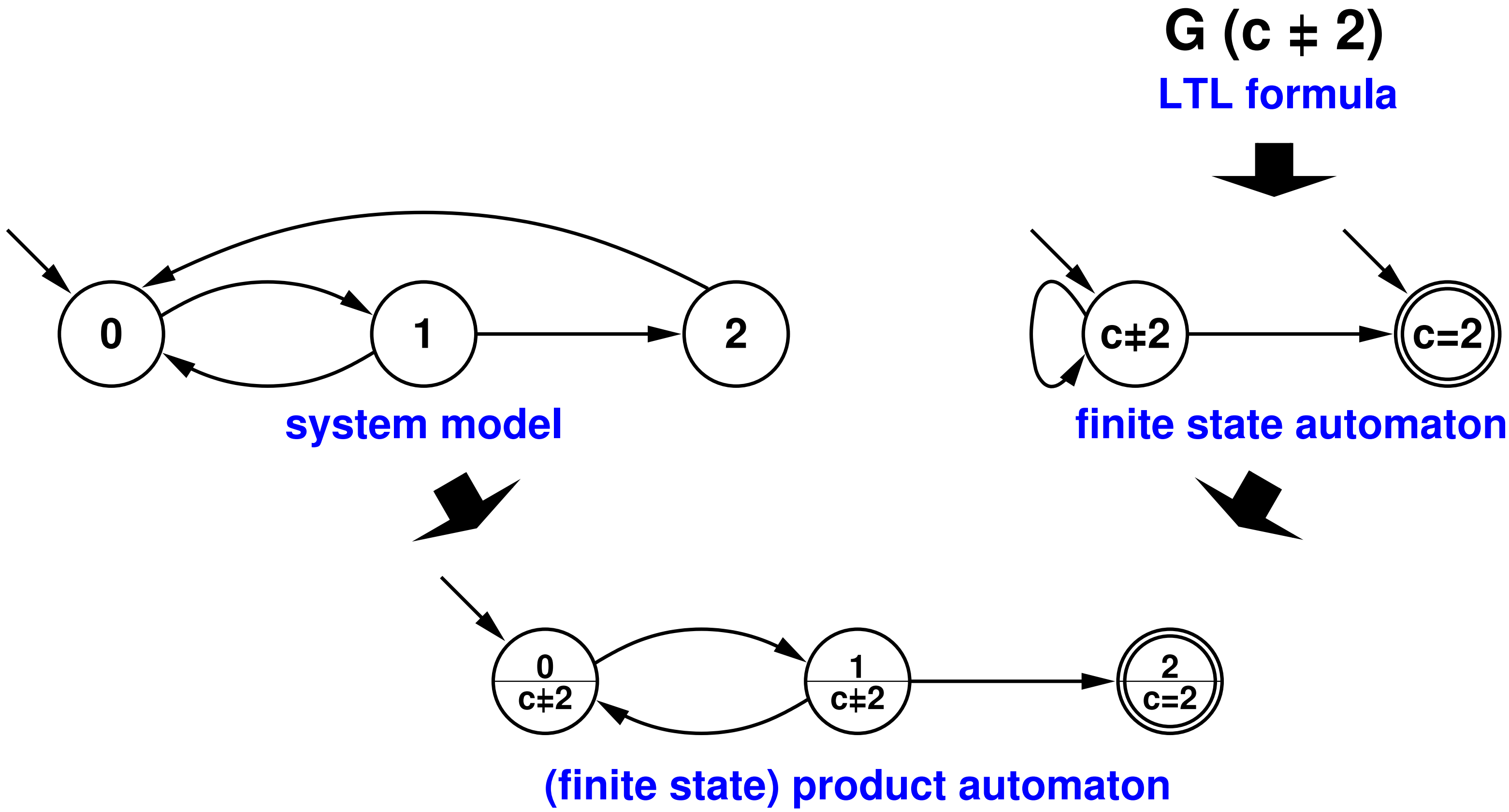
Liveness

“Something good will eventually happen.”

It remains possible for the “good thing” to occur.

Model Checking of Safety Properties

[Kupferman, Vardi '01]

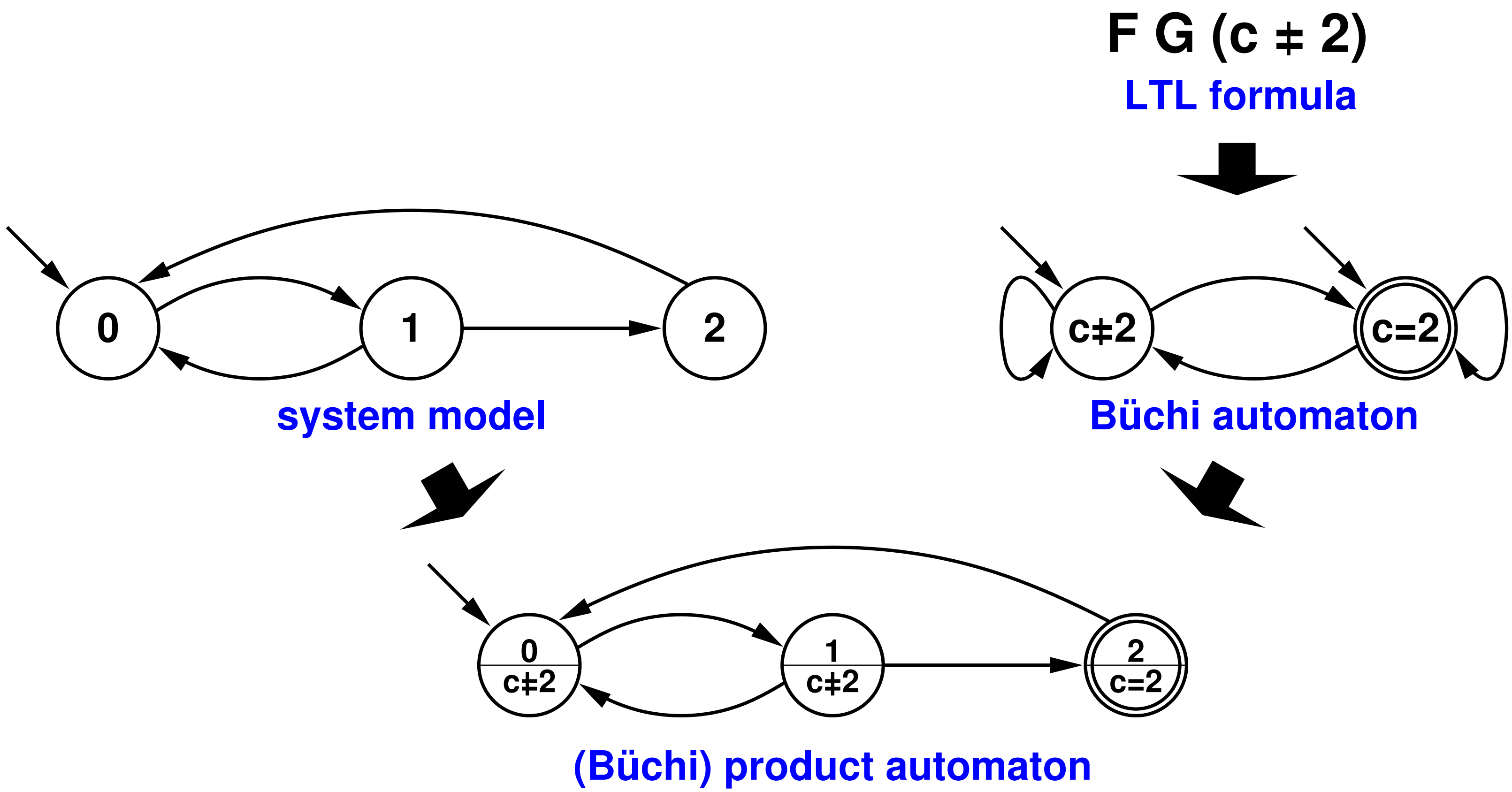


Property is false iff a bad state is reachable.

$\Rightarrow$ Find shortest finite path to bad state.

Model Checking of Liveness Properties

[Vardi, Wolper '86]



Property is false iff there is an (infinite) fair path.

⇒ Find fair lasso.

Contents

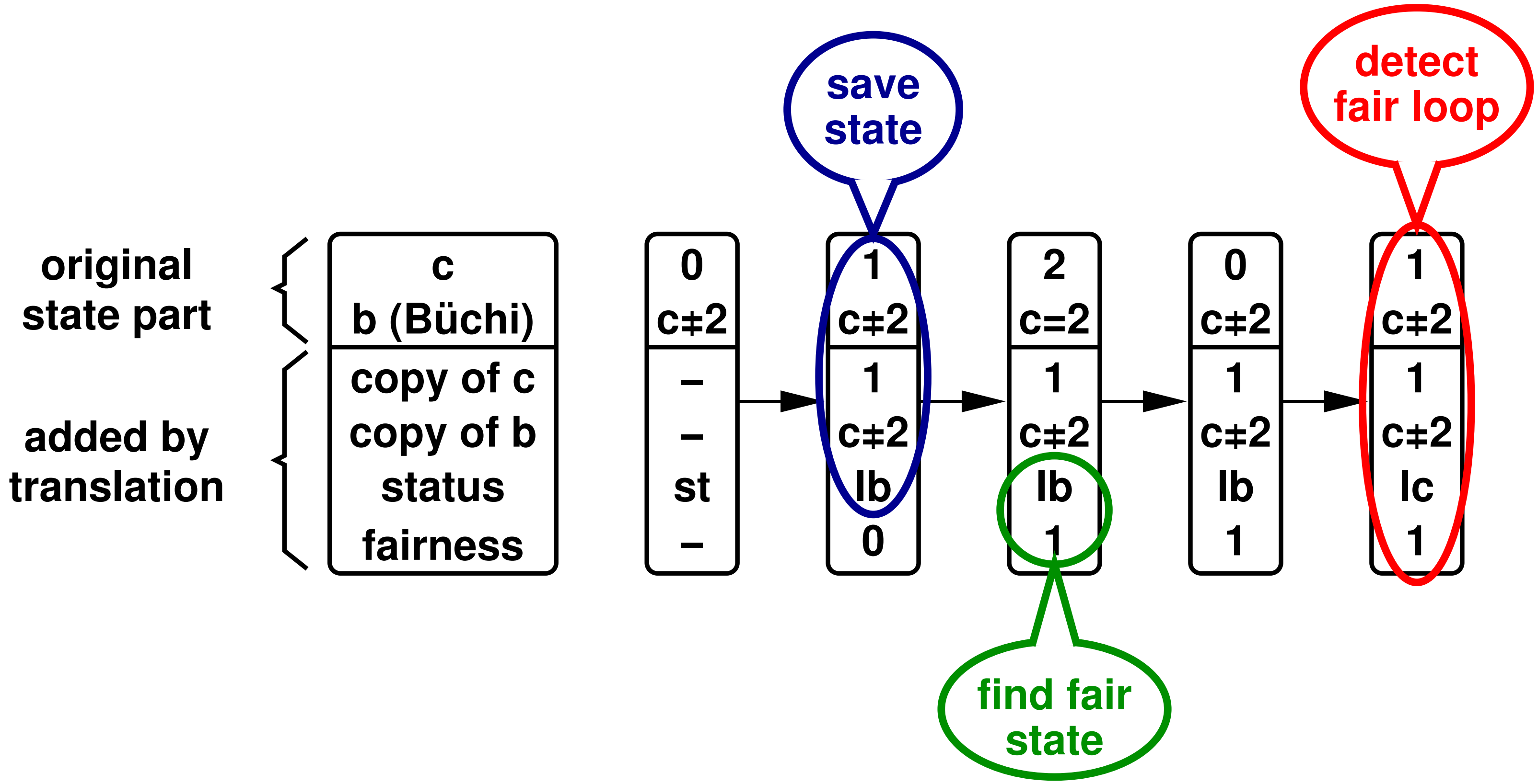
1. Model Checking 101

2. Liveness Checking as Safety Checking

3. Tight Büchi Automata

4. Conclusions

Liveness Checking as Safety Checking



State-recording translation:

1. Guess loop start: save current state.

2. Find fair state in loop.

3. Find second occurrence of saved state.

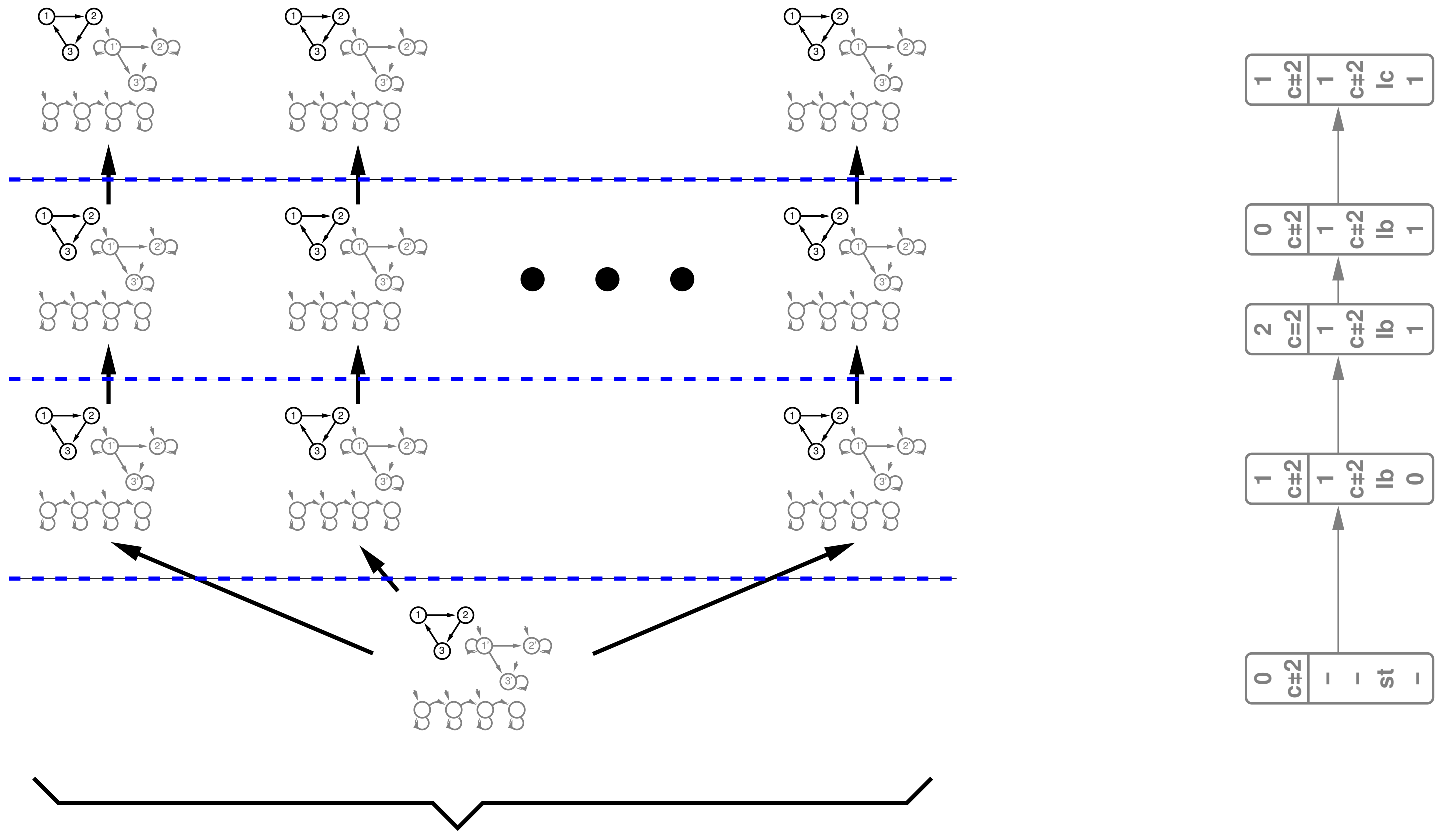
Complexity

loop closed

loop body, fair

loop body, not fair

stem



|S| branches,

no changing between branches

$$
\begin{aligned}
|S^{\mathbf{S}}| &= \mathbf{O}(|S|^2) & \qquad |T^{\mathbf{S}}| &= \mathbf{O}(|S| \cdot |T|) \\
r^{\mathbf{S}}, d^{\mathbf{S}} &= \mathbf{O}(d) & \qquad |(T^{\mathbf{S}})^*| &= \mathbf{O}(|S| \cdot |T^*|)
\end{aligned}
$$

Experiments

Show feasibility of model checking translated model: compare BDD-based symbolic model checking of LTL properties using

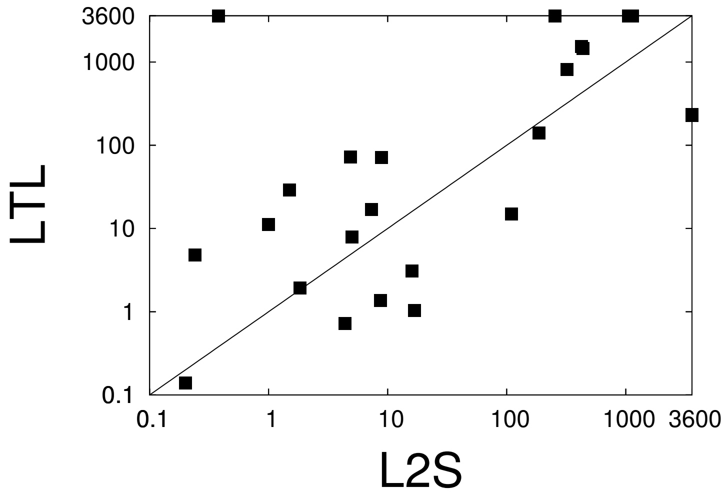
- Standard algorithm: NuSMV 2.2.2, labeled LTL
- Translated model: invariant checking in NuSMV 2.2.2, labeled L2S

Remarks

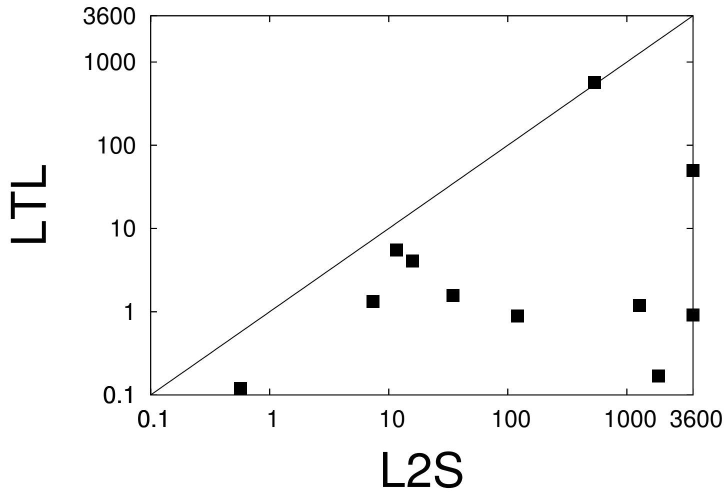
- LTL to Büchi automata with NuSMV's ltl2smv
- No cone of influence reduction
- BDD variable order:
  - Use static order if available
  - No dynamic reordering
  - Interleave original state variables and L2S copies

Results

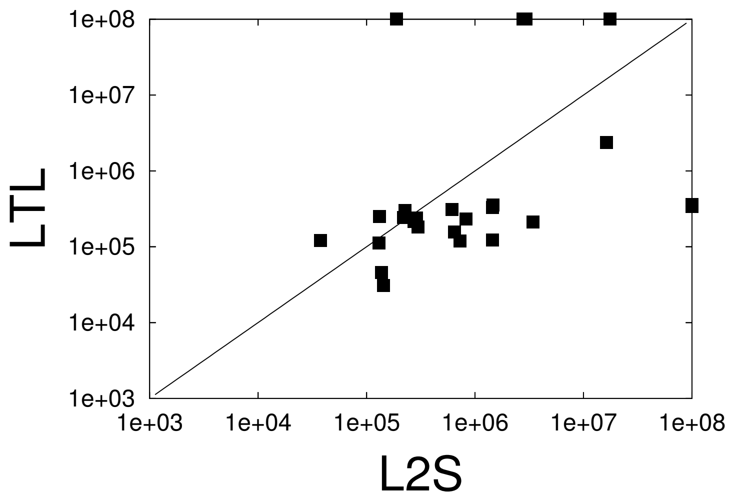
CPU time [seconds] — false



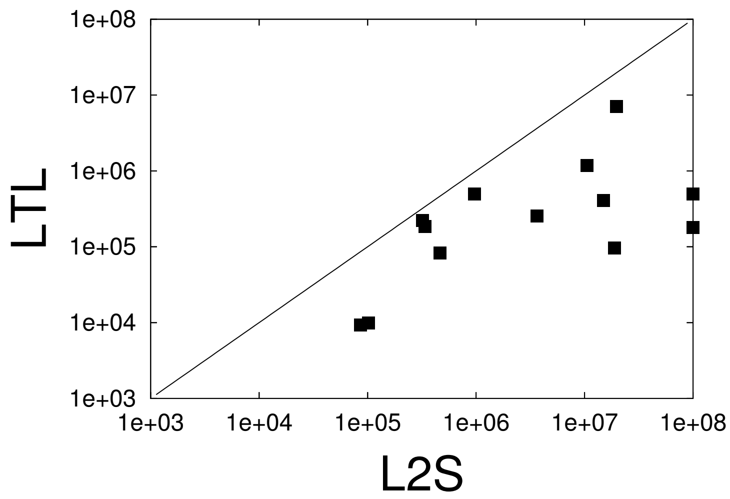
CPU time [seconds] — true



Memory [# BDD nodes] — false



Memory [# BDD nodes] — true



Half-way Summary

Benefits

- Find shortest lassos with a BDD-based model checker
- Make tools and methods for safety available for liveness properties
- Have quick and dirty liveness algorithm
- Need fewer liveness proofs

What's more

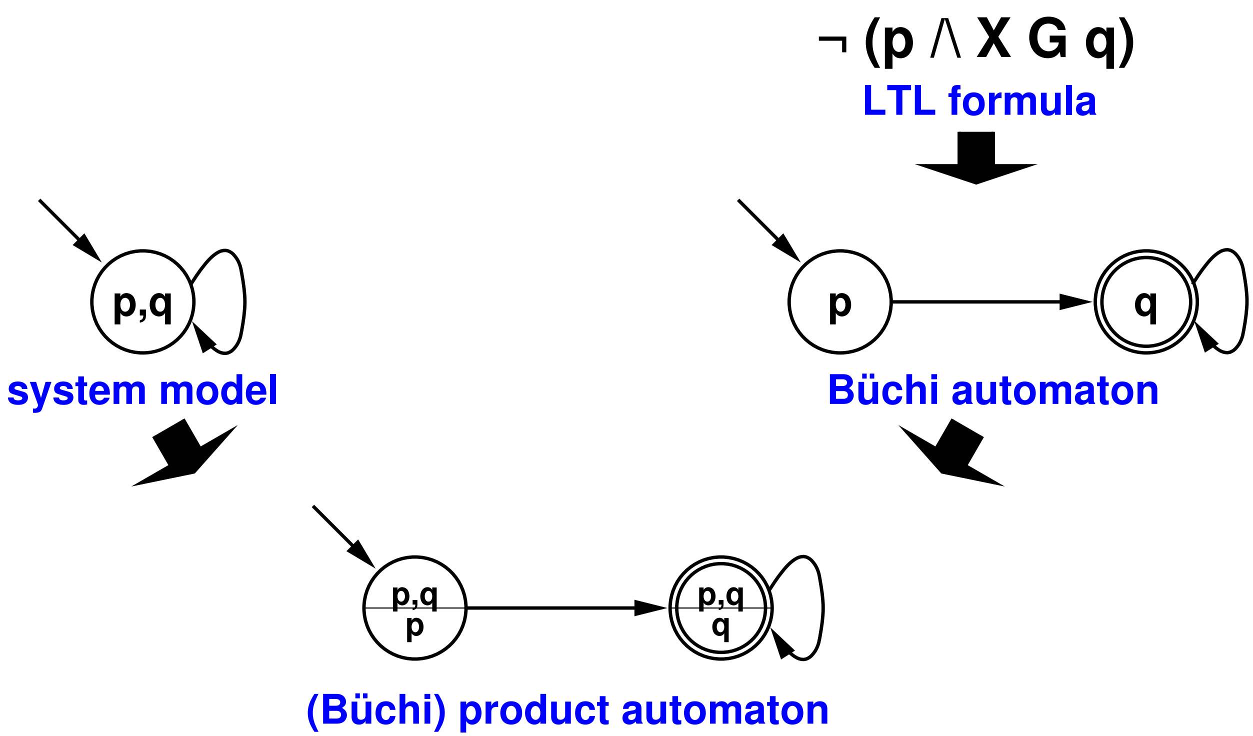
- Exponential speed up on selected examples
- Extension to infinite state systems:
  regular model checking, pushdown systems, timed automata
- Optimizations

Contents

1. Model Checking 101
2. Liveness Checking as Safety Checking
3. Tight Büchi Automata
4. Conclusions

Tight Büchi Automata

Not all Büchi automata allow to find shortest counterexamples:



To find shortest counterexamples, for each counterexample the Büchi automaton must have an accepting run of the same shape as the counterexample:

$$\forall \alpha = \beta\gamma^{\omega} \in Lang(B) \;.\; \exists \rho = \sigma\tau^{\omega} \in Runs(B) \;.\; \rho \models \alpha \wedge |\beta| = |\sigma| \wedge |\tau| = |\gamma|$$

⇒ Extend notion of tight automaton [Kupferman, Vardi '01] to Büchi aut.

How Bad Is It?

Let

- ϕ be a future time/mixed future and past time LTL property,
- $B^{\neg\phi}$ be a Büchi automaton constructed with the method of Gerth et al./Kesten et al., and
- $\alpha = \beta\gamma^{\omega}$ be a counterexample to ϕ.

Then there is an accepting run $\rho = \sigma\tau^{\omega}$ on α in $B^{\neg\phi}$ with

$$|\sigma| \leq |\beta| + (h_{f/p}(\phi) + 1)|\gamma|$$

and

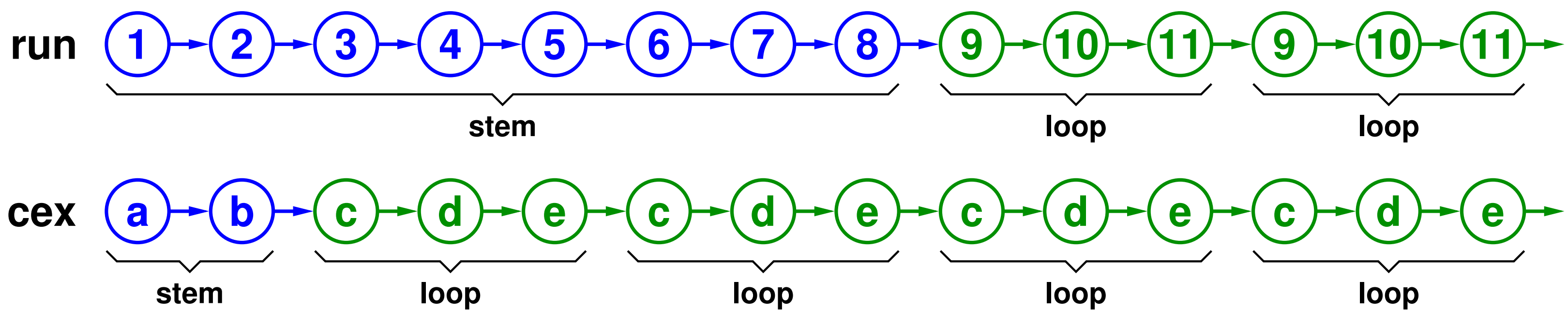
$$|\tau| = |\gamma|$$

where $h_{f/p}$ is the maximum number of nested future/past operators.

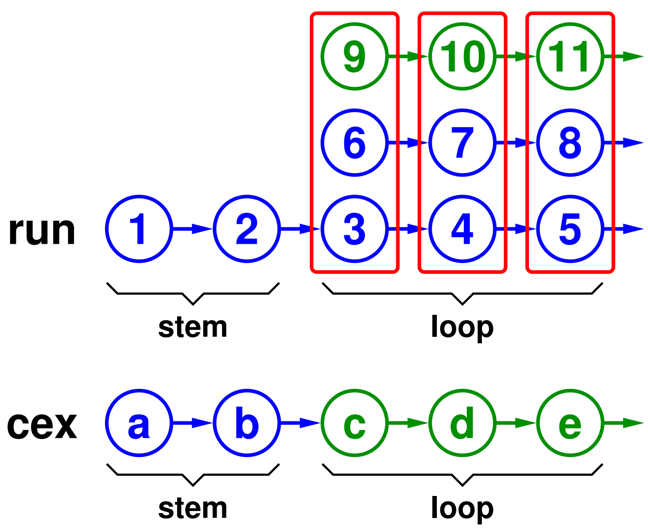
Popular methods to construct Büchi automata may lead to counterexamples with excess length linear in the maximum number of nested operators.

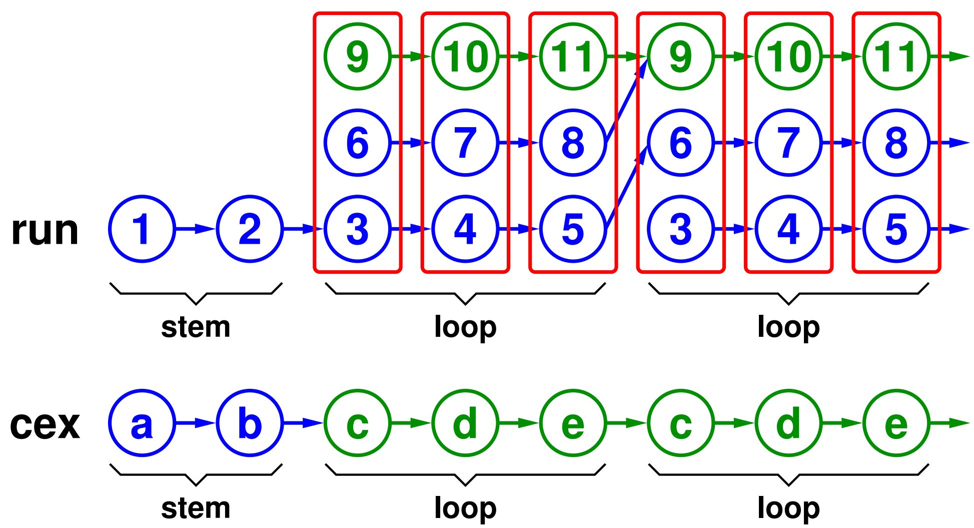
The method by Kesten et al. produces tight automata for future time LTL.

Assume the following (abstract) run and counterexample:



Have different parts of run work in parallel: form vectors of states





Experiments

Determine counterexample length using

– standard algorithm and standard automaton
– invariant checking of translated model and standard automaton
– invariant checking of translated model and tight automaton

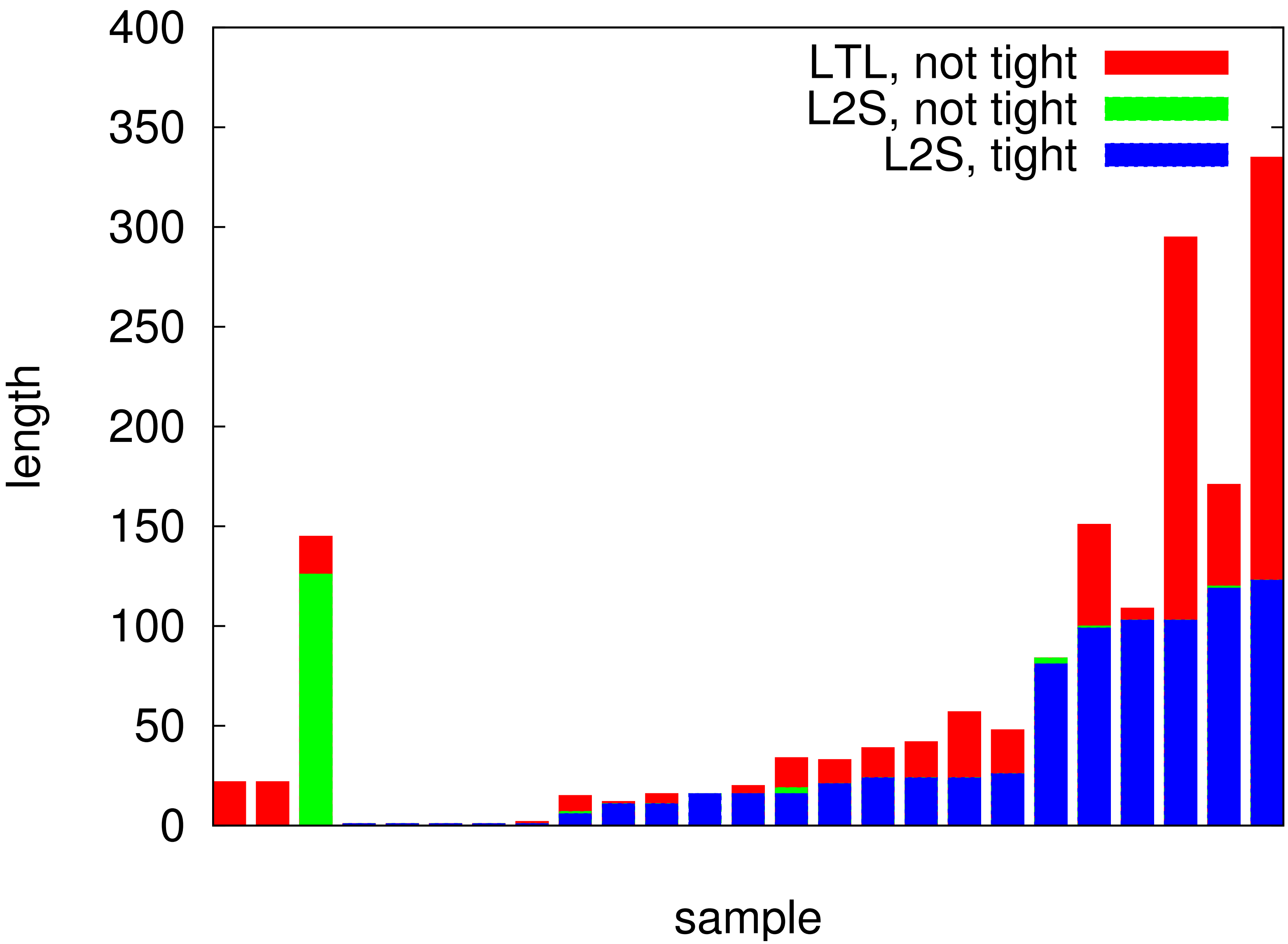
Compare finding shortest counterexamples with tight encoding using

– SAT-based BMC [Heljanko, Junttila, Latvala '05]
  $\Rightarrow$ preliminary incremental implementation of [Latvala et al. '05]
  modified NuSMV 2.2.2, labeled BMC
– BDD-based invariant checking of translated model, labeled L2S

Remarks

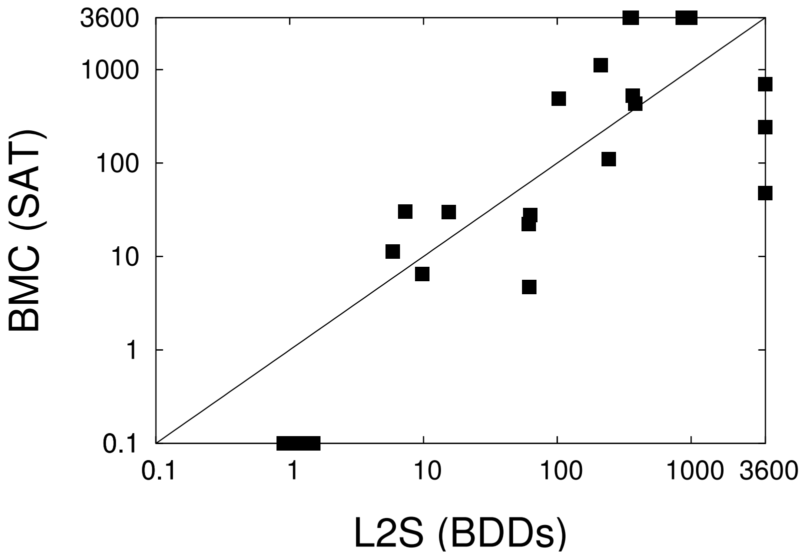
– as before, but
– no static order for BDDs (other than interleaving of original and L2S copies of state variables)

Results: Reduction in Counterexample Length

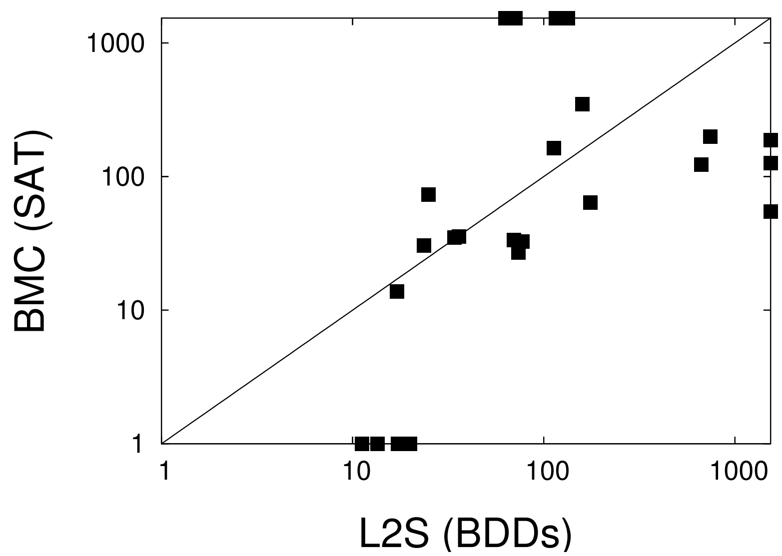


Results: BDDs vs. SAT

L2S vs incremental BMC
– CPU time [seconds]



L2S vs incremental BMC
– Memory [MByte]



Related Work

Liveness Checking as Safety Checking:

Shilov, Yi, Eo, O, Choe '01/'05 Reduction of SOEPDL ($\rhd$ 2M of C. Stirling) to reachability. Requires closure under Cartesian product and subset constructions. More powerful but doubly exponential.

Burch '90 Reduction for timed trace structures. Requires user to come up with appropriate time constraint.

Ultes-Nitsche '02 Satisfaction within fairness corresponds to some safety property. May change semantics.

Tight Büchi Automata:

Kupferman, Vardi '01 Shortest counterexamples for safety properties. Tight automata on finite words.

Benedetti, Cimatti '03 Virtual unrolling for BMC.

Latvala, Biere, Heljanko, Junttila '05 Inspiration for tight Büchi automata.

The End

Summary:

– Feasible translation from liveness to safety

– Tight Büchi automata

– Practical BDD-based method to find shortest counterexamples for LTL

Future Work:

– More powerful logics

– Tight Büchi automata for explicit state model checking

– Complementary property of tightness