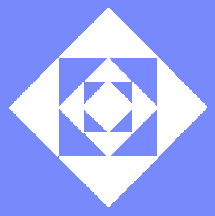


An Optimized Symbolic Bounded Model Checking Engine

Rachel Tzoref, Mark Matusevich, Eli Berger and Ilan Beer

© 2003 IBM Corporation

Agenda

- Bounded Model Checking
- The Symbolic Simulation Based Approach
- Optimizations
- Under-approximation
- Experimental Results

© 2003 IBM Corporation

The Bounded Model Checking Problem

- Given a Model M, a specification φ and a cycle bound k, determine whether φ holds in the first k cycles of M.
- Used mainly for falsification.
- Considered to be easier than full model checking.

© 2003 IBM Corporation

Different Approaches for Solving the Bounded Model Checking Problem

- The traditional approach, as in SAT-based BMC, is to analyze the behavior of the model based on the state variables.
  - Build a formula in which the variables are the state variables of the model, duplicated for each cycle.
  - Find a satisfying assignment to those variables, which violates the specification φ.
- Some symbolic simulation methods use an alternative approach: Analyze the behavior of the model as a function of the inputs to the model along all cycles (non-deterministic choices are treated as inputs).

© 2003 IBM Corporation

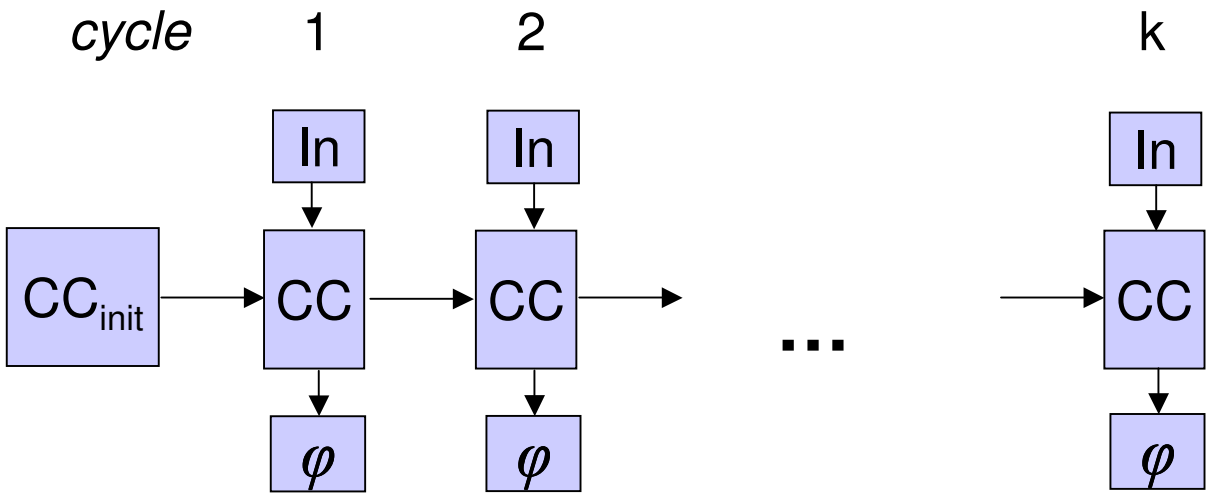
Agenda

- Bounded Model Checking
- The Symbolic Simulation Based Approach
- Optimizations
- Under-approximation
- Experimental Results

© 2003 IBM Corporation

Analyzing the Model as a Function of the Non-determinism

- Unroll the sequential model M and the specification φ into an iterative combinational circuit.
- Represent all nondeterministic signals in M by free inputs.
- In each cycle, build the BDD of φ as a function of the free inputs:



© 2003 IBM Corporation

Advantages and Drawbacks

- Advantages:
  - Insensitive to the number of state variables in the model.
  - Allows evaluating multiple properties in a single run, without repeating calculations.
- Drawbacks:
  - Sensitive to the amount of non-determinism in the model.
  - In each cycle, non-determinism is added, and therefore computation complexity grows as cycles advance.
- Potentially suitable for wide circuits and shallow bugs, for example:
  - Datapath
  - Equivalence Checking

© 2003 IBM Corporation

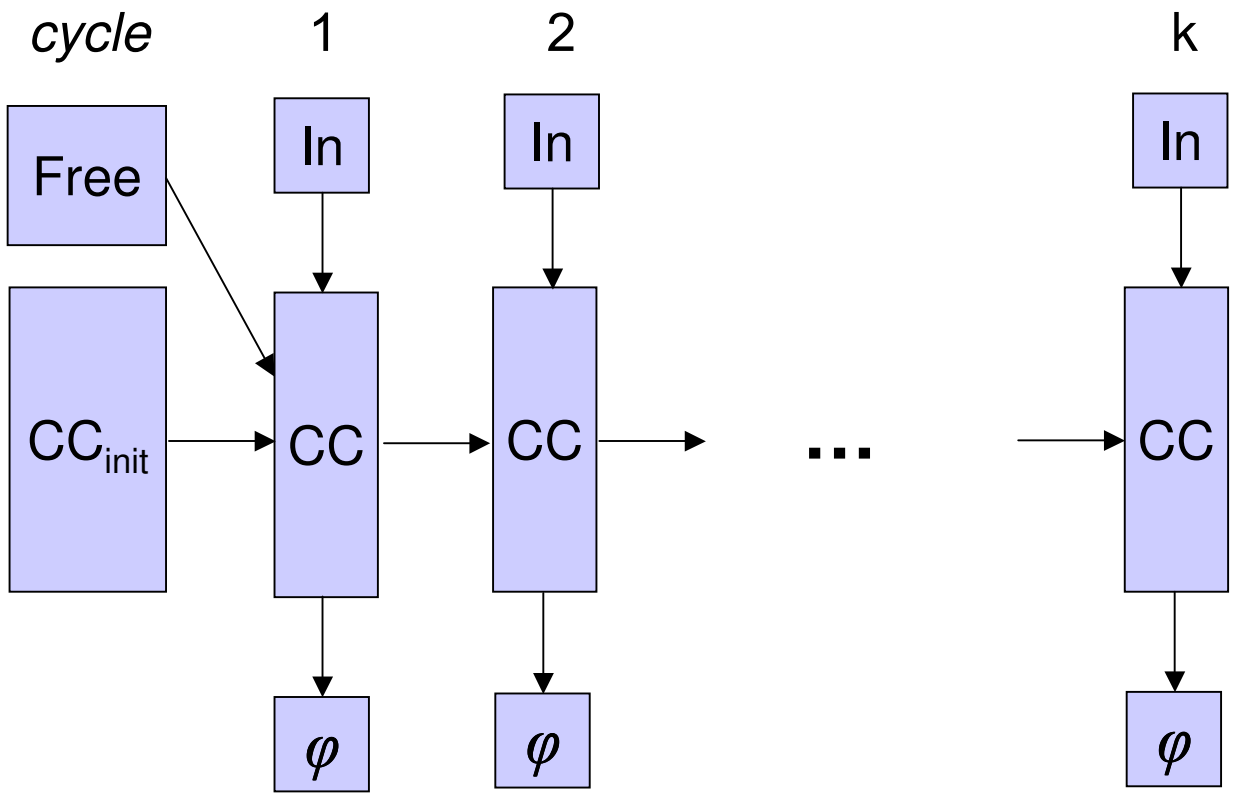
Agenda

- Bounded Model Checking
- The Symbolic Simulation Based Approach
- Optimizations
- Under-approximation
- Experimental Results

© 2003 IBM Corporation

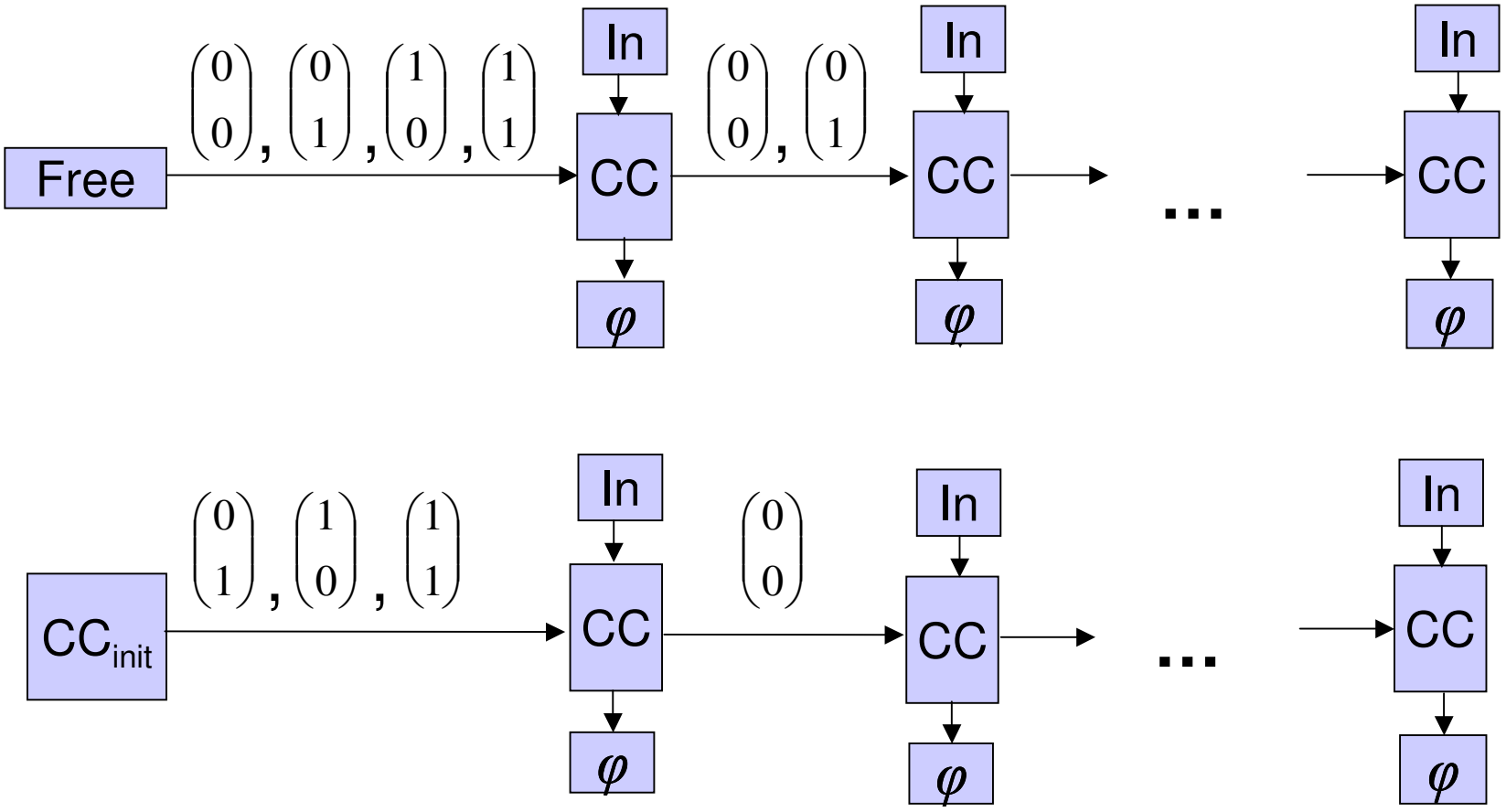
Open Machine

- Replace combinational logic that generates initial states by free inputs.



© 2003 IBM Corporation

Open Machine as Abstraction



© 2003 IBM Corporation

Open Machine as Abstraction - Summary

- Two levels of abstraction:
  - Each cycle of the open machine is an abstraction of its next cycle.
  - Each cycle of the open machine is an abstraction of the same cycle of the original machine.

© 2003 IBM Corporation

Using the Open Machine

- We can use this abstraction in order to simplify the original machine.
  - Constant propagation
    - If gate g is constant b in cycle i of the open machine, then:
      - for all j>=i gate g is constant b in cycle j of the open machine.
      - for all j>=i gate g is constant b in cycle j of the original machine.
  - Logical Equivalence
    - If gate g in cycle i has the same function as gate h in cycle j, then:
      - for all k>=0 gate g in cycle i+k has the same function as gate h in cycle j+k of the open machine.
      - for all k>=0 gate g in cycle i+k has the same function as gate h in cycle j+k of the original machine.

© 2003 IBM Corporation

Representing the Iterative Combinational Circuit

- Option 1: First unroll the circuit k times, then start evaluating the properties.
  - Enables ‘global’ heuristics, such as evaluating properties according to evaluation complexity rather than according to cycle order.
- Option 2: At each stage, hold the sub-circuit, required to evaluate current cycle.
  - Allows evaluating the properties without complete circuit unrolling.
  - Enables reductions reducing memory consumption.
- We managed to benefit from both options, by developing a data structure that:
  - Represents both the circuit and the open machine unrolled to k cycles.
  - Internally, represents all replications of a gate by a single object.
  - Consequently, enables efficient implementation of constant propagation and logical equivalence.

© 2003 IBM Corporation

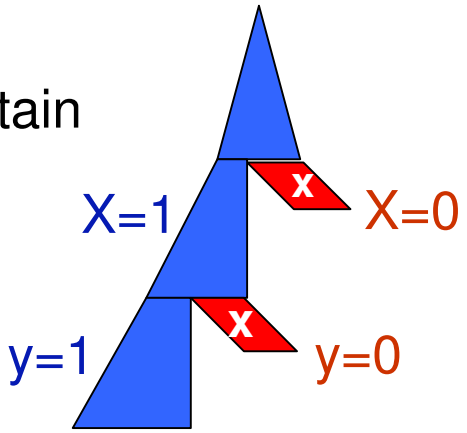
Agenda

- Bounded Model Checking
- The Symbolic Simulation Based Approach
- Optimizations
- Under-approximation
- Experimental Results

© 2003 IBM Corporation

Applying Under-approximation

- Search only a subset of the search space.
- Intuitive under-approximation: Set a free input at a certain cycle to a constant value.

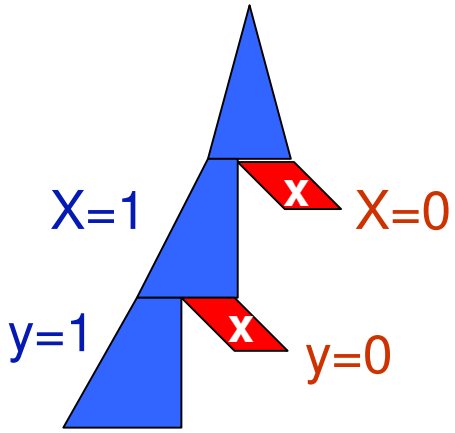


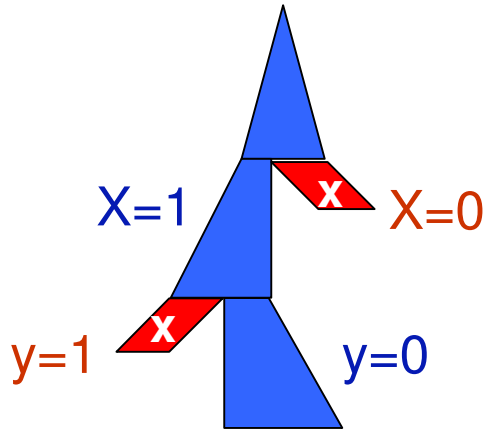
- Advantages:
  - Scalable: tradeoff between BDD size and coverage.
  - The computation never explodes. If needed, a stronger under-approximation is used.
  - Degenerate model behavior only at a specific cycle.

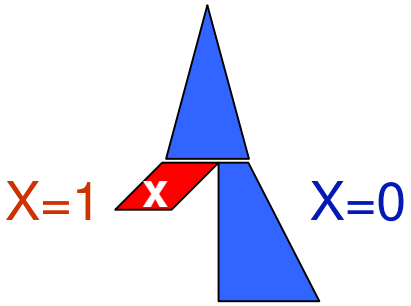
© 2003 IBM Corporation

Exact Evaluation with Under-approximation

- We implemented a mode that combines under-approximation with backtracking in order to cover parts of the search space that were skipped as a result of previous under-approximations.
- When reaching the cycle bound, we backtrack and start over computation using a different set of choices for the reduced variables.
- Thus, we will eventually achieve a full coverage of the search space.







© 2003 IBM Corporation

Agenda

- Bounded Model Checking
- The Symbolic Simulation Based Approach
- Optimizations
- Under-approximation
- Experimental Results

© 2003 IBM Corporation

Experimental Results

# inputs	*# vars*	*# props*	*Optimized BDD-Based BMC*			*zChaff Based SAT Solver*		
			Time (sec.)	Mem (MB)	Cycles	Time (sec.)	Mem (MB)	Cycles
4	279	1	63	353	49	957	169	49
32	363	15	1948	606	100	out	571	70
58	202	2	33	136	6,7	out	9	0
175	1124	1	21529	816	100	out	937	35
39	377	1	77	176	23	415	207	23
112	375	1	17	96	10	10	24	10
14	109	1	43	74	16	23	19	16

© 2003 IBM Corporation