Bounded Model Checking (BMC)

Dr Olga Tveretina

Summer semester, 2009

Background: model checking

- ***Given:***
  - a finite transition system *M*
  - a property *P* (in some temporal logic)
- ***The model checking problem:***
  - Does P holds in M?

$$\mathbf{M} \stackrel{?}{\models} \mathbf{P}$$

Temporal properties

- ***Safety* properties:**
  - "Always x=y"
  - G (x=y)
- ***Liveness* properties:**
  - "Reset can always be reached"
  - GF Reset
  - "From some point on, always switch_on"
  - FG switch_on

OBDDs and symbolic model checking

- OBDD is a canonical form to represent Boolean functions
- They are often more compact than 'traditional' normal forms as CNFs, DNFs and can be manipulated efficiently
- The reachable state-space is represented by a OBDD
- The property is evaluated recursively, by iterative fix point computations on the reachable state-space

Problems with OBDDs

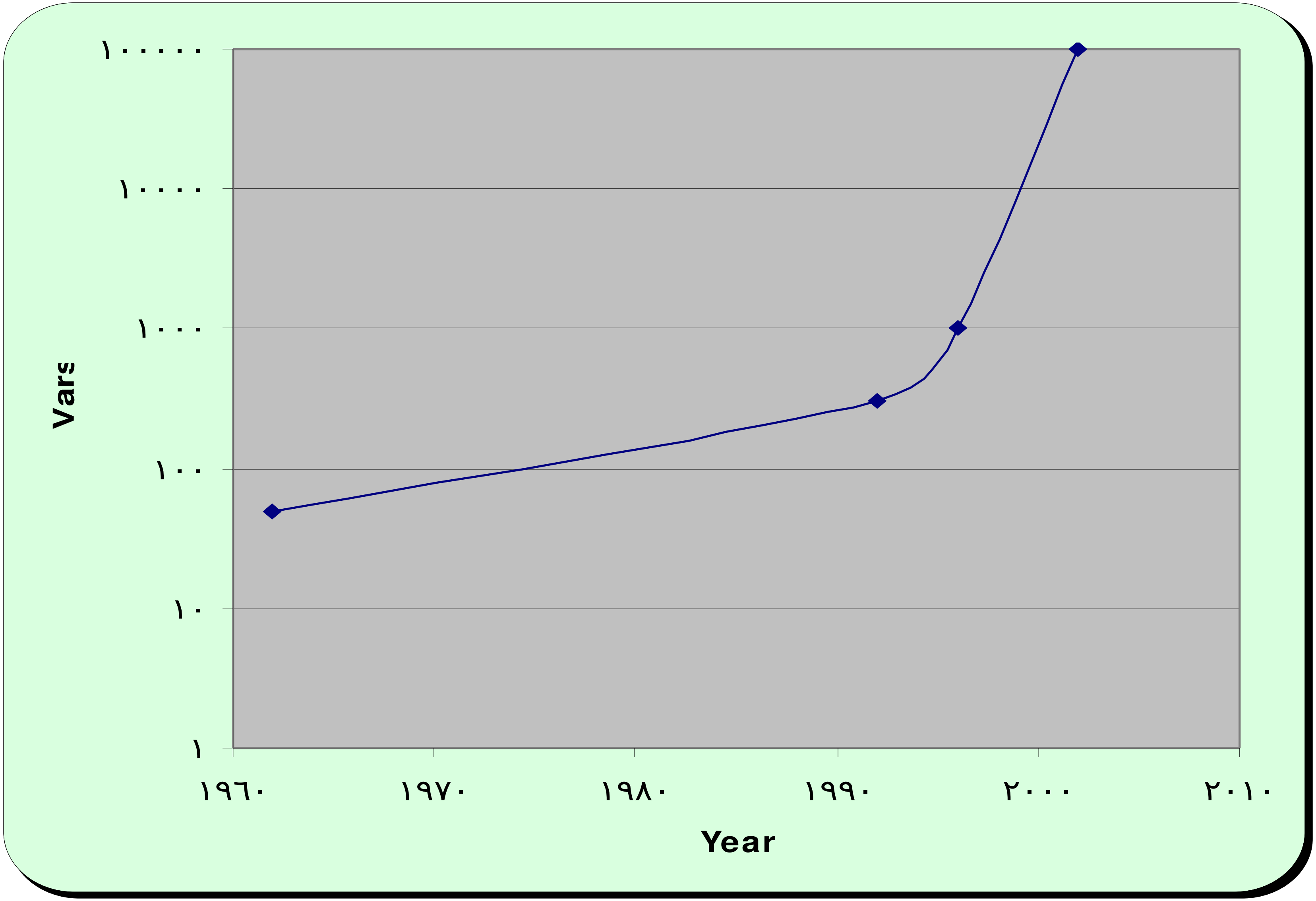
- BDDs are a canonical representation, but often become too large
- Variable ordering must be uniform along paths.
- Selecting right variable ordering very important to built small BDDs
  - time consuming or needs manual intervention
  - in some cases no space efficient variable ordering exists
- Alternative approaches to model checking use SAT procedures

Advantages of SAT procedures

- SAT procedures also operate on Boolean formulas but do not use canonical forms
- Do not suffer from the potential space explosion of BDDs
- Different orderings of variables are possible on different branches
- There exist very efficient implementations

SAT solver progress 1960 -2010

(E.Clarke)



Bounded model checking

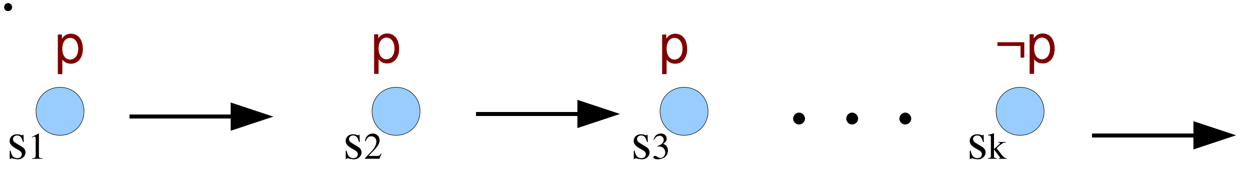
- A. Biere, A. Cimatti, E. Clarke, Y. Zhu, Symbolic Model Checking without BDDs, TACAS'99
- E. Clarke, A. Biere, R. Raimi, Y. Zhu, Bounded Model Checking Using Satisfiability Solving, 2001

Bounded model checking

- Based on SAT
- There is a counterexample of length k <=> propositional formula is satisfiable
- BMC for LTL reduced to SAT in poly time

Example:

- Most of the safety properties can be expressed as ***'always p'***, where p is a propositional variable
- Is there a state reachable within k steps that satisfies ¬p?



Bounded model checking

- Existential model checking problem M |= Ef for an LTL formula f and a Knipke structure M

- To look for a witness to the property that can be represented within a bound of k steps

- Given k, the problem is reduced to the satisfiability of a propositional formula $[[M,f]]_k$

- If $[[M,f]]_k$ is satisfiable then the propositional model provides a witness of k steps to f

Bounded model checking

- The method is not complete

- If $[[M,f]]_k$ is unsatisfiable then nothing can be said about the existence of a solution for M |= f models with higher bound

- The typical technique is to generate and solve $[[M,f]]_k$ for increasing values of k

Bounded model checking

- Effective and practical technique, especially in the process of falsification, i.e. bug funding
- Bounded model checking based on SAT procedures not BDD
- Smart DFS search of SAT potentially will get faster to a satisfying sequence (counterexample)
- No exponential space

Creation of propositional formula

- ***Given:***
  - a transition system M
  - a temporal logic formula f
  - a user-supplied bound k

- ***Construct:***
  - a propositional formula $[[M,f]]_k$ is satisfiable iff f is valid along some computation path of M

Creation of propositional formula

- For state transition system M and time bound k, the unrolled transition relation is

$$[[M]]_k = I(s_0) \wedge \bigwedge_{i=0}^{k-1} T(s_i, s_{i+1})$$

  - $I(s_0)$ is the characteristic function of the set of initial states
  - $T(s_i, s_{i+1})$ is the characteristic function of the transition relation

- a propositional formula $[[M,f]]_k$ is satisfiable iff f is valid along some computation path of M

Creation of propositional formula

Example:

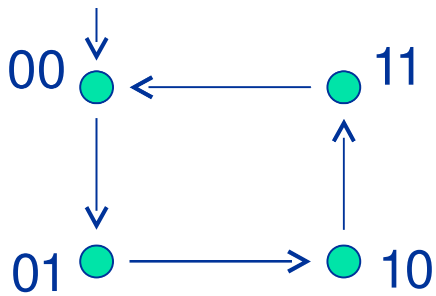
- Consider the CTL formula **EF p**
- Check whether **EF p** can be verified in two time steps, i.e. k=2

$$[[M,f]]_2 = I(s_0) \land T(s_0,s_1) \land T(s_1,s_2) \land (p(s0) \lor p(s1) \lor p(s2))$$

- Here, $(p(s0) \lor p(s1) \lor p(s2))$ is $[[\mathbf{EF}\ p]]_2$

Safety property example

2-bit counter: the least significant bit represented by a Boolean variable A and the most significant by B



Transition relation:

$$(A' \longleftrightarrow \neg A) \wedge (B' \longleftrightarrow A \lozenge B)$$

$\lozenge$ stands for XOR, $\longleftrightarrow$, XNOR

I(s0): $(\neg A0 \wedge \neg B0) \quad \wedge$

T(s0,s1): $((A1 \longleftrightarrow \neg A0) \wedge (B1 \longleftrightarrow (A0 \lozenge B0))) \quad \wedge$

T(s1,s2): $((A2 \longleftrightarrow \neg A1) \wedge (B2 \longleftrightarrow (A1 \lozenge B1))) \quad \wedge$

p(s0): $(\ A0 \wedge B0 \quad \vee$

p(s1): $A1 \wedge B1 \quad \vee$

p(s2): $A2 \wedge B2 \ )$

Liveness property example

- We add a transition from state (1,0) back to itself

- Define:

inc(s,s')=(A' <—> ¬A) ∧ (B' <—> (A ◊ B))
T(s,s')=inc(s,s') ∨ (B ∧ ¬A ∧ B' ∧ ¬A')

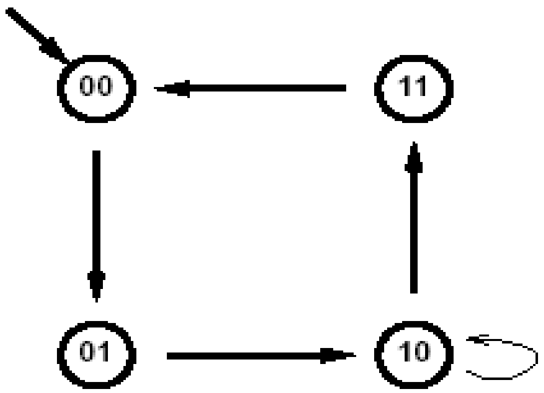


Fig. 1. A two-bit counter with an extra transition

- A counter must eventually reach state (1,1): **AF** (b ∧ a)

- A counterexample that demonstrates this would be a path starting at the initial state, in which the counter never reaches state (1,1): **EG** p, where p = ¬B ∨ ¬A

Liveness property example

- Set the time bound k for checking EG p at 2
- All candidate paths will then have k+1, or 3 states, an initial one and two reached upon two successive transitions: s0, s1, s2
- The transition relation must hold for k=2

$$[[M]]_2 = I(s_0) \wedge T(s_0,s_1) \wedge T(s_1,s_2)$$

- The sequence of states s0, s1, s2 must be a part of a loop, i.e.

$$T(s2,s3) \wedge (s3=s0 \vee s3=s1 \vee s3=s2)$$

Liveness property example

I(s0): $(\neg A0 \wedge \neg\, B0) \quad \wedge$

T(s0,s1): $((A1 \longleftrightarrow \neg A0) \wedge (B1 \longleftrightarrow (A0 \lozenge B0)) \quad \vee$

$B1 \wedge \neg A1 \wedge B0 \wedge \neg A0) \quad \wedge$

T(s1,s2): $((A2 \longleftrightarrow \neg A1) \wedge (B2 \longleftrightarrow (A1 \lozenge B1)) \quad \vee$

$B2 \wedge \neg A2 \wedge B1 \wedge \neg A1) \quad \wedge$

T(s2,s3): $((A3 \longleftrightarrow \neg A2) \wedge (B3 \longleftrightarrow (A2 \lozenge B2)) \quad \vee$

$B2 \wedge \neg A2 \wedge B1 \wedge \neg A1) \quad \wedge$

s3=s0: $(\quad (A3 \longleftrightarrow A0) \wedge (B3 \longleftrightarrow B0) \quad \vee$

s3=s1: $(A3 \longleftrightarrow A1) \wedge (B3 \longleftrightarrow B1) \quad \vee$

s3=s2: $(A3 \longleftrightarrow A2) \wedge (B3 \longleftrightarrow B2) \quad) \quad \vee$

p(s0): $(\quad \neg A0 \wedge \neg B0 \quad \vee$

p(s1): $\neg A1 \wedge \neg B1 \quad \vee$

p(s2): $\neg A2 \wedge \neg B2 \quad)$

Liveness property example

- The formula is satisfiable
- The satisfying assignment corresponds to a path from initial state (0,0) to (0,1) and then to (1,0) followed by the self-loop at state (1,0), and is a counterexample to **AF** (B ∧ A)
- Removing the self-loop would remove the lines

$$\mathbf{B_i} \wedge \neg\mathbf{A_i} \wedge \mathbf{B_{i-1}} \wedge \neg\mathbf{A_{i-1}}$$

- The formula then become unsatisfiable

Determining the bound k

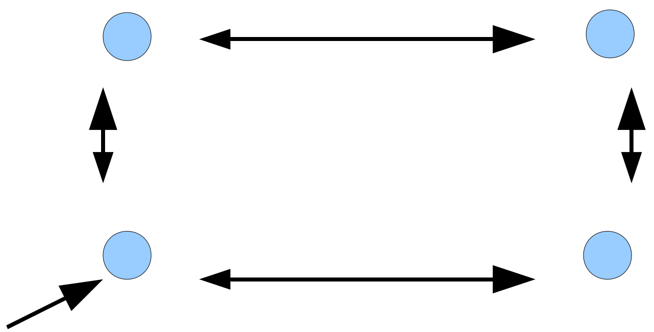
- For every model M and LTL property P there exists k such that

$$M \models_k P \quad \longrightarrow \quad M \models P$$

- The minimal such k is the completeness threshold (CT)

Determining the bound k

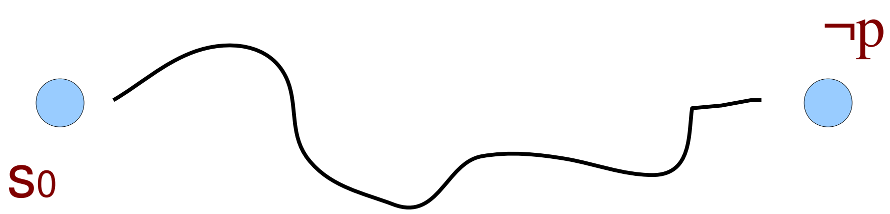
- Diameter d = longest 'shortest path' from an initial state to any other reachable state
- Recurrence diameter rd = longest loop-free path
- rd ≥ d



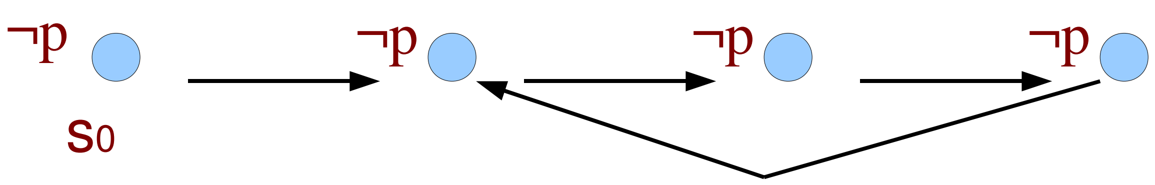
d = 2
rd = 3

Determining the bound k

- ***Theorem:*** For **G**p properties CT = d.



- ***Theorem:*** For **F**p properties CT = rd.



- ***Open problem:*** The value of CT for general LTL logic is unknown.

What BMC useful for?

- A.I. planing problems:
  Can we reach a desirable state in k steps?
- Verification of safety properties:
  Can we find a bad state in k steps?
- Verification:
  Can we find a counterexample in k steps?

BMS vs. MC

- ***Advantages of BMS:***
  - Counterexamples – found faster and of minimal length
  - Less space, no manual intervention (order on variables for OBDDs)
  - The modern SAT solvers are very efficient
- ***Disadvantages of BMS:***
  - With the limit k, completeness cannot be always achieved

BMC

- A model checker called BMC has been implemented, based on bounded model checking.
- It's input language is a subset of the SMV language.
- It takes in a circuit description, a property to be proven, and a user supplied time bound k.
- It then generates the propositional formula.