LTL Generalized Model Checking

Revisited

Patrice Godefroid

Microsoft Research

Nir Piterman

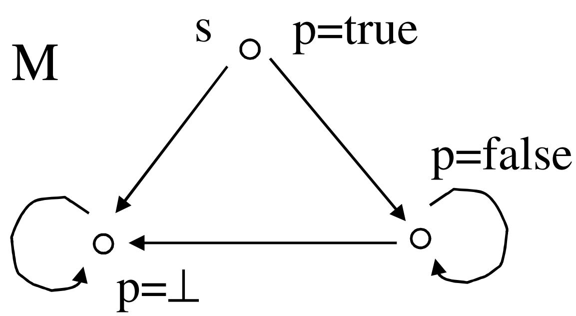
Imperial College

Model Checking via Automatic Abstraction

- Implemented in software model checkers like SLAM, BLAST,…
- Traditional iterative abstraction procedure:
  1. Abstract:
     - generate a finite abstraction A from the concrete program C such that A simulates C (using predicate abstraction, theorem proving)
  2. Check:
     - given any universal temporal-logic formula f, compute [A |= f] :
       if [A |= f] = true, then return true (we then know [C |= f] = true)
  3. Refine:
     - Otherwise ([A |= f] = false), refine A, then go to Step 1
     - Ex: with predicate abstraction, add predicates to refine the model A
- Limitations:
  - Restricted to universal properties (no existential properties)
  - [A |= f] = false does not imply anything about C
  - Could the analysis be more precise for an acceptable cost?

A Solution: 3-Valued Models and Logics

- Richer models A that distinguish what is true/false/unknown of C
  - Ex: “partial Kripke structure” [Fitting92, Bruns-G99]



  - Ex: “Modal Transition System” (may/must trans.) [Larsen+88]
  - These formalisms are all equally expressive [G-Jagadeesan03]

- Reasoning about 3-val. models requires 3-val. temp. logic
  - Ex: [(M,s) |= p] = true, [(M,s) |= AXp] = false, [(M,s) |= EXp] = $\perp$

- Complexity of 3-valued MC = complexity of MC [Bruns-G00]

New Abstract-Check-Refine Process

- New automatic-abstraction procedure: (3 improvements) [G-Huth-Jagadeesan01,…]
  1. Abstract: generate a 3-valued abstraction A from the concrete program C that preserves *true, false, unknown* properties of C (same cost)
     – Formally, A « C with the abstraction preorder « on 3-valued models
  2. Check: given any temporal-logic formula f,
     - (3-valued model checking) compute [A |= f] : (same cost)

       if [A |= f] = true or false, then return true or false (respectively)
     - Otherwise (generalized model checking)

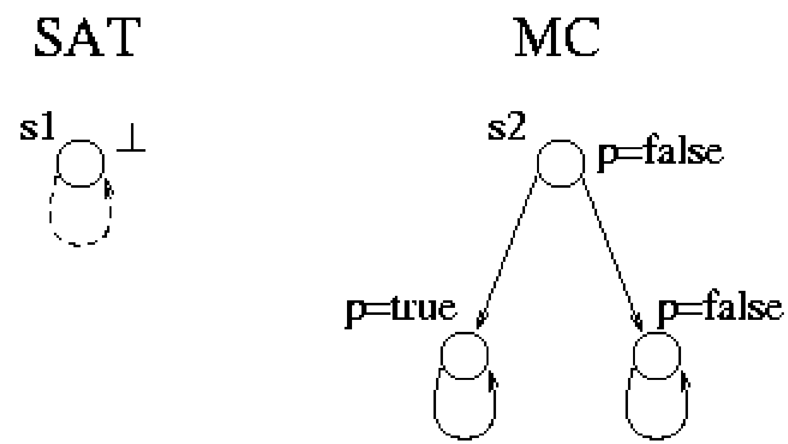
       -if there is no concretization C of A such that C satisfies f, ret false

       -if there is no concretization C of A such that C violates f, ret true
  3. Refine: Otherwise, refine A, then go to Step 1

Generalized Model Checking (GMC)

- Definition: [Bruns-G00]
  Given a program abstraction A and a temporal logic formula f, does there exist a concretization C of A such that C satisfies f?

- GMC is a generalization of both
  - Satisfiability (SAT)
  - Model Checking (MC)



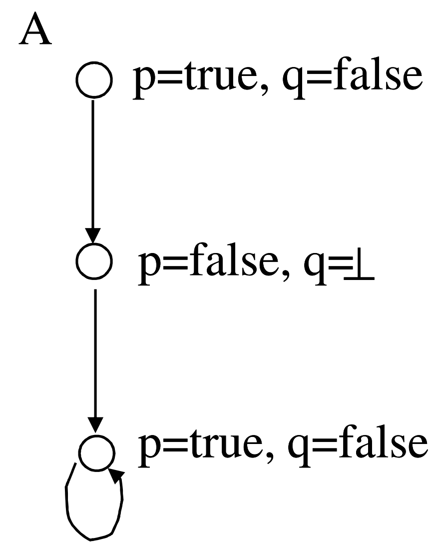
- GMC can be more expensive than MC (includes SAT)
  - in |f| (but worst-case and f is usually short) [Bruns-G00]
  - in |A| (polynomial) but linear for e.g. safety properties [G-J02]

- GMC can also be more precise than MC…

Ex where GMC is more precise than MC

```
Program P( ) {
 int x,y = 1,0;
 x,y = 2*f(x), f(y);
 x,y = 1,0;
}
```

$\xrightarrow[\text{p: "x is odd"} \quad \text{q: "y is odd"}]{\text{Predicate abstraction}}$



Property "(eventually y is odd) and (always, x is odd or y is even)"

is represented by the LTL formula $f = F(q) \wedge G(p \vee \neg q)$

$MC(A,f) = \perp$ …but $GMC(A,f) = false$!

What is the complexity of GMC?

- [Bruns-Godefroid00]:
  - Branching-Time Logics: GMC has the same complexity as SAT
  - Linear-Time Logics: GMC is harder than SAT and MC

Logic	MC	SAT	GMC
PL	Linear	NP-Complete	NP-Complete
PML	Linear	PSPACE-Complete	PSPACE-Complete
CTL	Linear	EXPTIME-Complete	EXPTIME-Complete
μ-calculus	NP$\cap$co-NP	EXPTIME-Complete	EXPTIME-Complete
LTL	PSPACE-Complete	PSPACE-Complete	~~EXPTIME-Complete~~

Wrong!

- This paper: 2EXPTIME-complete !

New Result: LTL GMC is 2EXPTIME-compl.

- New upper bound: given a PKS M and a LTL formula f,
  - Translate f into a NBW A (exponential blow-up)
  - Translate A into a DPW A' (exponential blow-up) (*)
  - Combine M with A' to get a APW A'' over 1-letter alphabet
  - Check that L(A'') is non-empty (polynomial time)
  - **Theorem:** L(A'') is non-empty iff GMC(M,f) = true
  - Note: in [BG00], step (*) is missing and the direct ABW A'' construction is wrong as L(A'') is underapproximate

- New lower-bound:
  - **Theorem:** GMC for LTL is 2EXPTIME-hard
  - Proof: reduction from 2EXPTIME-hard LTL realizability problem [Pnueli-Rosner89]

New: Linear Completeness Preorder

- The previous results are for the traditional abstraction preorder « on 3-valued models: a « c implies
  - a =< c: For all p, L(a,p) =< L(c,p)

    where ⊥ =< true, ⊥ =< false, true =< true, false =< false, ⊥ =< ⊥
  - For every a->a', there exists c->c' such that a' « c'
  - For every c->c', there exists a->a' such that a' « c'
- New linear completeness preorder: a $«_L$ c implies
  - For every w in L(a), there exists w' in L(c) such that w =< w'
  - For every w' in L(c), there exists w in L(a) such that w =< w'
- **Theorem:** a $«_L$ c iff (for all f in LTL: [a |= f] =< [c |=f])
  - 3-valued LTL logically characterizes $«_L$

GMC for LTL with $\ll_L$

- For f in LTL, GMC(M, f, $\ll_L$) is defined as
  - Does there exists M' such that M $\ll_L$ M' and [M' |= f] = true ?

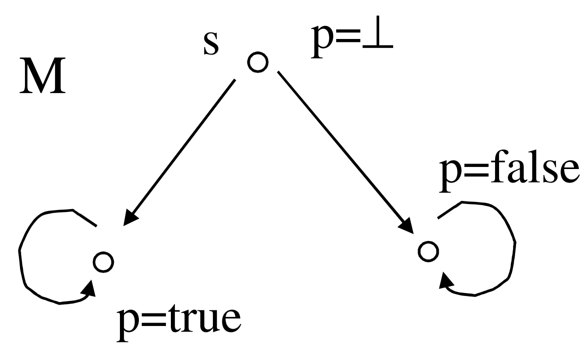
- **Theorem:** GMC(M, f, $\ll_L$) is EXPSPACE-complete
  - Upper bound: translate f into a NBW A (exponential blow-up),
  - build a 3-valued NBW A' such that
    w in L(A') iff there is w =< w' and [w' |= f] = true,
  - check L(M) ⊆ L(A') (space logarithmic in |M| and polynomial in |A'| [SistlaVardiWolper87], hence space exponential in |f|)
  - Lower bound: reduction from EXPSPACE-hard tiling problem [vanEmdeBoas97]

- Thus, GMC(M, f, $\ll_L$) is "only" EXPSPACE-complete (vs. 2EXPTIME-complete) and requires only space log in |M|!

Comparing « and «$_L$

- **Theorem:** for any LTL formula f,
  - M « M' implies M «$_L$ M',
  - hence GMC(M,f) = true implies GMC(M, f, «$_L$) = true

- The opposite is not true:
  - LTL $f = (p \wedge Xp) \vee (\neg p \wedge \neg Xp)$
  - $[s \models f] = \perp$
  - GMC(s, f, «$_L$) = true
  - but GMC(s, f) = false !



- GMC(M, f, «$_L$) is weaker (and cheaper) than GMC(M, f)

Model Complexity of GMC

- GMC(M, f, $\ll_L$) requires only logarithmic space in |M|
- but GMC(M, f) is polynomial (PTIME-complete) in |M|
  - The degree of the polynomial depends on the DPW A_f for f
- **Theorem:**
  - LTL GMC(M,f) is linear in |M| for weak (incl. safety) properties
    - Proof: the DPW for f is then a DWW, and the product with M is a 1-letter-alphabet AWW, whose emptiness can be checked in lin time
  - LTL GMC(M,f) is quadratic in |M| for response, persistence and generalized reactivity[1] properties
    - Proof: the DPW for is then a DBW, DCW or DPW with 3-priorities, and the product with M is a 1-letter-alphabet ABW, ACW or APW with 3-priorities, whose emptiness can be checked in quadratic time

Conclusions: LTL GMC Revisited

- LTL GMC(M,f) is 2EXPTIME-complete
  - instead of EXPTIME-complete [BG00]
- New linear completeness preorder $\ll_L$
- GMC(M, f, $\ll_L$) is only EXPSPACE-complete
- and requires only logarithmic space in |M|
- While GMC(M,f) is polynomial (PTIME-complete) in |M|
- but only linear or quadratic in |M| in many cases
  - linear for safety and weak properties
  - quadratic for response, persistence, generalized reactivity[1]