Inexact Arithmetic in Model Checking of DTMCs

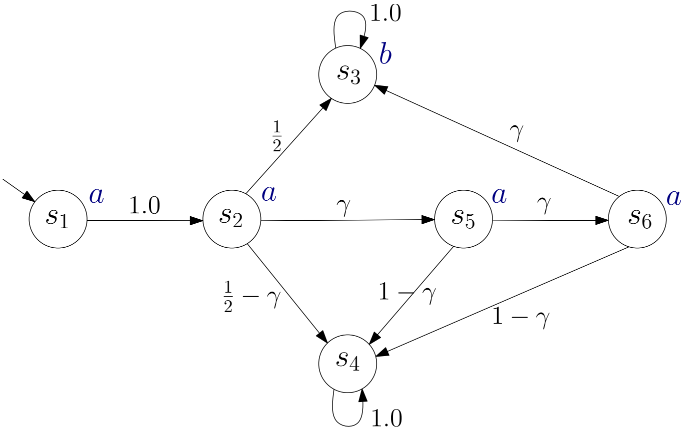
Two Decades of Probabilistic Verification
—Reflections and Perspectives—

Ralf Wimmer, Bernd Becker

Albert-Ludwigs-University Freiburg, Germany

November 14^{th}, 2007

A DTMC

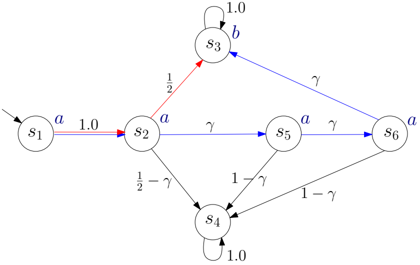


Probability to compute:

$$P^{?}\big(c\ U\ P_{\leq \frac{1}{2}}(a\ U\ b)\big)$$

Computing probabilities

Let γ be a small constant $< \frac{1}{2}$.



State	$P^{?}(a\,U\,b)$	$P_{\leq \frac{1}{2}}(a\,U\,b)$ satisfied?
s_1	$\frac{1}{2} + \gamma^3$	no
s_2	$\frac{1}{2} + \gamma^3$	no
s_3	1	no
s_4	0	yes
s_5	γ^2	yes
s_6	γ	yes

PRISM

Let's see, what PRISM says for $\gamma = 10^{-6}$:

```
probabilistic
const double gamma = 0.000001;

module sys
 s: [1..6] init 1;

 [] s=1 -> 1.0: (s'=2);
 [] s=2 -> 0.5: (s'=3) + gamma: (s'=5) + (0.5-gamma): (s'=4);
 [] s=3 -> 1.0: (s'=3);
 [] s=4 -> 1.0: (s'=4);
 [] s=5 -> gamma: (s'=6) + (1-gamma): (s'=4);
 [] s=6 -> gamma: (s'=3) + (1-gamma): (s'=4);
endmodule
```

Result:

```
yes = 5, no = 1, maybe = 0
Time for model checking: 0.022 seconds.
Result: 1.0
```

MRMC

What does MRMC say (for $\gamma = 10^{-6}$)?

Transitions:

```
STATES 6
TRANSITIONS 10
1 2 1.0
2 3 0.5
2 4 0.499999
2 5 0.000001
3 3 1.0
4 4 1.0
5 4 0.999999
5 6 0.000001
6 3 0.000001
6 4 0.999999
```

Labels:

```
#DECLARATION
a b c
#END
1 a
2 a
3 b
5 a
6 a
```

Result:

```
$RESULT: ( 1.0000000, 1.0000000, 0.0000000, 1.0000000, 1.0000000, 1.0000000 )
$STATE: { 1, 2, 4, 5, 6 }
```

The origin of the problem

- The model checker has to represent the value $\frac{1}{2} + \gamma^3$ such that it is larger than 0.5.

The origin of the problem

- The model checker has to represent the value $\frac{1}{2} + \gamma^3$ such that it is larger than 0.5.
- For $\gamma = 10^{-6}$ this value is

$$\frac{1}{2} + 10^{-18} = 0.500000000000000001$$

The origin of the problem

- The model checker has to represent the value $\frac{1}{2} + \gamma^3$ such that it is larger than 0.5.
- For $\gamma = 10^{-6}$ this value is

$$\frac{1}{2} + 10^{-18} = 0.500000000000000001$$

- Floating-point arithmetic with 64 bit can represent numbers with an accuracy of about 15 decimal digits.

The origin of the problem

- The model checker has to represent the value $\frac{1}{2} + \gamma^3$ such that it is larger than 0.5.
- For $\gamma = 10^{-6}$ this value is

$$\frac{1}{2} + 10^{-18} = 0.50000000000000001$$

- Floating-point arithmetic with 64 bit can represent numbers with an accuracy of about 15 decimal digits.
- The number is rounded downwards to 0.5.

The origin of the problem

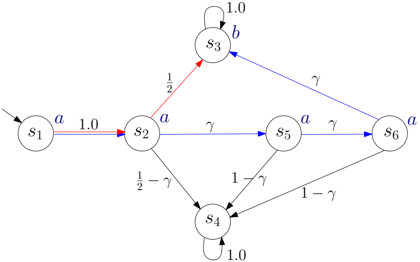
- The model checker has to represent the value $\frac{1}{2} + \gamma^3$ such that it is larger than 0.5.
- For $\gamma = 10^{-6}$ this value is

$$\frac{1}{2} + 10^{-18} = 0.500000000000000001$$

- Floating-point arithmetic with 64 bit can represent numbers with an accuracy of about 15 decimal digits.
- The number is rounded downwards to 0.5.
- Changing the value of $\frac{1}{2} + \gamma^3$ to 0.5 flips the truth value of $P_{\leq 0.5}(a\, U\, b)$ in s_1 and s_2 from "no" to "yes".

Computing probabilities – inexact arithmetic

Let γ be a small constant $< \frac{1}{2}$.



State	$P^{?}(a\ U\ b)$	$P_{\leq \frac{1}{2}}(a\ U\ b)$ satisfied?
s_1	$\frac{1}{2} + \gamma^3$	yes
s_2	$\frac{1}{2} + \gamma^3$	yes
s_3	1	no
s_4	0	yes
s_5	γ^2	yes
s_6	γ	yes

How to solve this problem?

- Using exact arithmetic?
  - Slow and memory consuming
- Using interval arithmetic with save rounding?
  - Result is sometimes (often?) "unknown".
- Computing certificates testifying that the result is correct?
  - Not always applicable ...

We are not alone . . .

Reliable results are also a hot topic in other communities:

- SAT-Solvers:
  - Certificates for unsatisfiability (resolution trees)
- QBF-Solvers:
  - Certificates for SAT and UNSAT (??)
- Linear Programming:
  - Certificates for UNSAT (Farkas-Lemma)
  - Exact computation with inexact arithmetic?

Literature (1)

Conrado Daws.
Symbolic and Parametric Model Checking of Discrete-time Markov Chains
ICTAC 2004 (LNCS Vol. 3407)

Reduce model checking for DTMCs to the evaluation of regular expressions.

+ Probabilities can be adjusted *after* the construction of the regular expression
+ Easy to use exact arithmetic (only addition and multiplication needed)
− No nested PCTL-formulae
− Scalability?? (exprensive computation of regular expressions)

Literature (2)

Tingting Han, Jost-Pieter Katoen.
Counterexamples in Probabilistic Model Checking
TACAS 2007 (LNCS Vol. 4424)

Compute counterexamples for PCTL-formulae $P_{\leq p}(a\, U\, b)$ using a shortest path algorithm.

+ Optimal counterexamples (minimal number of paths + most probable paths)
– Scalability?? (Explicit representation!)