SMT Solvers, CBMC

http://d3s.mff.cuni.cz

Department of Distributed and Dependable Systems

Pavel Parízek

CHARLES UNIVERSITY IN PRAGUE

faculty of mathematics and physics

Z3

- SMT solver
  - theories: linear integer arithmetic, uninterpreted functions, arrays, bit vectors, ...
  - Input: SMT-LIB v2 (http://smtlib.cs.uiowa.edu/)
- Created by Microsoft Research
- Supports both Windows and Linux
- Source code & wiki: https://github.com/Z3Prover/z3
- Online interface: http://riseforfun.com/Z3

Department of Distributed and Dependable Systems

SMT-LIB: basics

```
; comment: after the semicolon until the end of a line
;

; specify the logic to be used
(set-logic QF_UFLIA)

; condition that should hold
(assert (= c (+ a 2)))          ; c == a + 2
```

SMT-LIB: expressions

```
(not P)
(and b1 b2)
(or ...)
(xor ...)

(+ a b c d)
(= a b)
(=> true false)
```

SMT-LIB: predicates and functions

```
; function symbol
(declare-fun Plus (Int Int) Int)

; predicate
(declare-fun Odd (Int) Bool)

; constant
(declare-fun a () Int)
(declare-const c Int)  ; syntactic sugar
```

Department of Distributed and Dependable Systems D3S

SMT-LIB: example

```
(declare-const a Int)
(declare-const b Int)
(declare-const c Int)

(assert (<= a b))
(assert (<= b c))
(assert (<= c a))
(assert (= 6 (+ a b c)))

; result: sat, unsat, unknown
(check-sat)

(get-model)
```

Department of Distributed and Dependable Systems D3S

How to encode programs using SMT

```
int max(int a, int b) {
  int r;
  if (a < b) {
    r = b;
  }
  else { // a >= b
    r = a;
  }
  assert (a <= r && b <= r);
  return r;
}
```

How to encode programs using SMT

```
int max(int a, int b) {
  int r;
  if (a < b) {
    r = b;
  }
  else { // a >= b
    r = a;
  }
  assert (a <= r && b <= r);
  return r;
}
```

```
a = *, b = *
r = *
if (a < b) {
  r = b;
}
else { // a >= b
  r = a;
}

assert (a <= r && b <= r);
```

How to encode programs using SMT

```
a = *, b = *
r = *
if (a < b) {
  r = b;
}
else { // a >= b
  r = a;
}

assert (a <= r && b <= r);
```

```
(declare-const a Int)
(declare-const b Int)
(declare-const r Int)

(assert (or
  (and (< a b) (= r b))
  (and (>= a b) (= r a))
))

(assert (and (<= a r) (<= b r)))

(check-sat)
(get-model)
```

Variable assignments

```
int x = 0;

x = x + 1;

assert (x == 1);
```

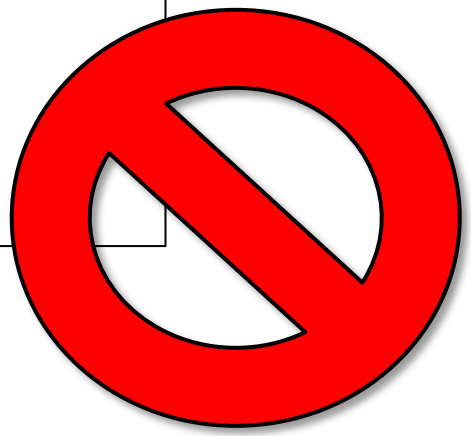
```
(declare-const x1 Int)

(assert (= x1 0))

(assert (= x1 (+ x1 1)))

(assert (= x1 1))

(check-sat)
(get-model)
```



Variable assignments: versions & SSA

- Variables have multiple versions
- Static single assignment (SSA)

```
int x = 0;

x = x + 1;

assert (x == 1);
```

```
(declare-const x1 Int)
(declare-const x2 Int)

(assert (= x1 0))

(assert (= x2 (+ x1 1)))

(assert (= x2 1))

(check-sat)
(get-model)
```

Department of Distributed and Dependable Systems D3S

Arrays

```
(declare-const c Int)

(declare-const a1 (Array Int Int))
(declare-const a2 (Array Int Int))

(assert (= c (select a1 10)))

(assert (= a2 (store a1 1 20)))
```

Department of Distributed and Dependable Systems D3S

Bounded model checking

- Often used with SAT (propositional logic)
  - **Q: Why?**
  - Limited expressiveness: no “+” and “-”
  - **Q: How to encode statements like x = a + b ?**

Integer addition in HW/CPU

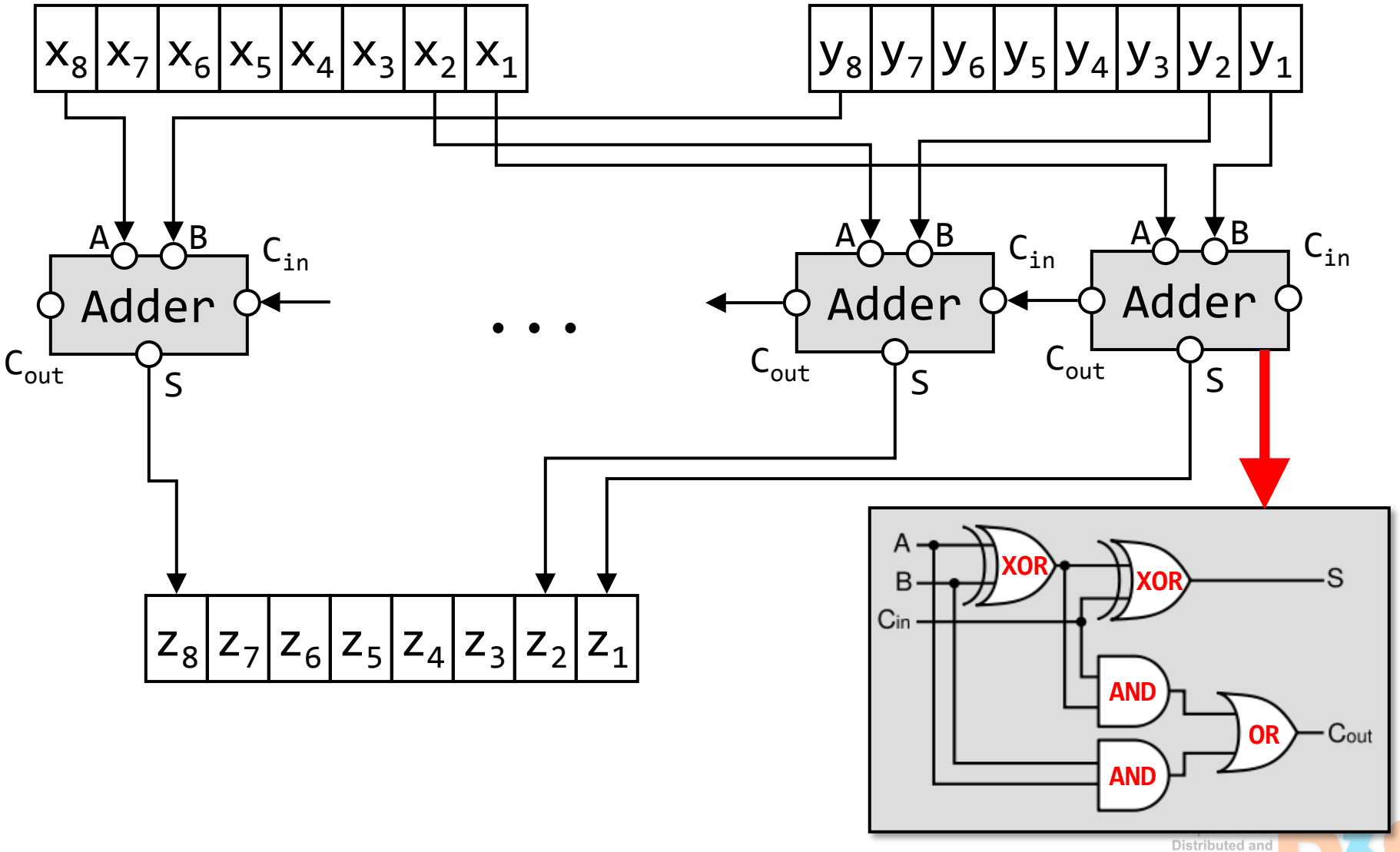


Image taken from Wikipedia

Encoding “Adder” as a SAT instance

0x! $((A \& B \& C_{in}) \Rightarrow (S \& C_{out})) \&$

1x! $((!A \& B \& C_{in}) \Rightarrow (!S \& C_{out})) \&$

1x! $((A \& !B \& C_{in}) \Rightarrow (!S \& C_{out})) \&$

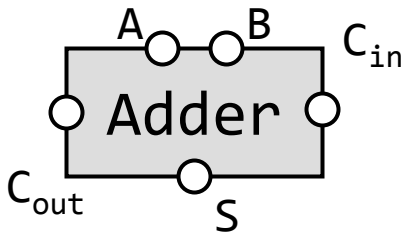
1x! $((A \& B \& !C_{in}) \Rightarrow (!S \& C_{out})) \&$

2x! $((!A \& !B \& C_{in}) \Rightarrow (S \& !C_{out})) \&$

2x! $((!A \& B \& !C_{in}) \Rightarrow (S \& !C_{out})) \&$

2x! $((A \& !B \& !C_{in}) \Rightarrow (S \& !C_{out})) \&$

3x! $((!A \& !B \& !C_{in}) \Rightarrow (!S \& !C_{out}))$



CBMC

- Bounded model checker for program in C/C++
- Developed at Oxford & Carnegie Mellon Uni
- http://www.cprover.org/cbmc/
- Source code and binaries freely available
  - Platforms: Windows, Linux, Mac OS

CBMC: how to use it

- Download
  - http://www.cprover.org/cbmc/download/cbmc-5-4-win.zip
- Run from the Visual Studio Command Prompt
  - Why: correctly initialized environment
- Examples
  - http://d3s.mff.cuni.cz/teaching/program_analysis_verification/files/bmc-examples.zip

Department of Distributed and Dependable Systems