Formal Checkers and Solvers for HW Design and Verification: Part II (SMT Solvers):

Ternary Propagation-Based Local Search for Bit-Precise Reasoning

---

Aina Niemetz
Stanford University

joint work with Mathias Preiner (Stanford University)

AHA (Virtual) Retreat, July 29-30, 2020

Satisfiability Modulo Theories (SMT)

Given a (quantifier-free) FOL formula and a combination of theories

$$((x \ll 01) \geq 00) \wedge (x < 01) \wedge ((\text{read}(\text{write}(a, x, x), x \cdot 10) = x + 01)$$

$x, 00, 01, 10 \ldots$ Bit-Vectors of size 2 $\quad$ $a \ldots$ Array

is there an assignment to x such that this formula evaluates to **true**?

Satisfiability Modulo Theories (SMT)

Given a (quantifier-free) FOL formula and a combination of theories

$$((x \ll 01) \geq 00) \wedge (x < 01) \wedge ((\text{read}(\text{write}(a, x, x), x \cdot 10) = x + 01)$$

$x, 00, 01, 10 \ldots$ Bit-Vectors of size 2 $\qquad$ $a \ldots$ Array

is there an assignment to x such that this formula evaluates to **true**?

No

Ongoing Challenges

- native support for **new theories**
  - strings
  - sequences
  - graphs
  - ...
- **performance and scalability**
  - SMT solvers typically at the **backend** of a tool chain
  - improvements propagate all the way up the stack
  - we keep pushing research frontiers

Theory of Fixed-Size Bit-Vectors

- constants, variables: $00000010,\ 2_{[8]},\ x_{[32]},\ y_{[2]}$
- bit-vector operators: $=,\ <,\ >,\ \sim,\ \&,\ \ll,\ \gg,\ \circ,\ [:],\ \ldots$
- arithmetic operators modulo 2^n (overflow semantics!)

Bit-Blasting

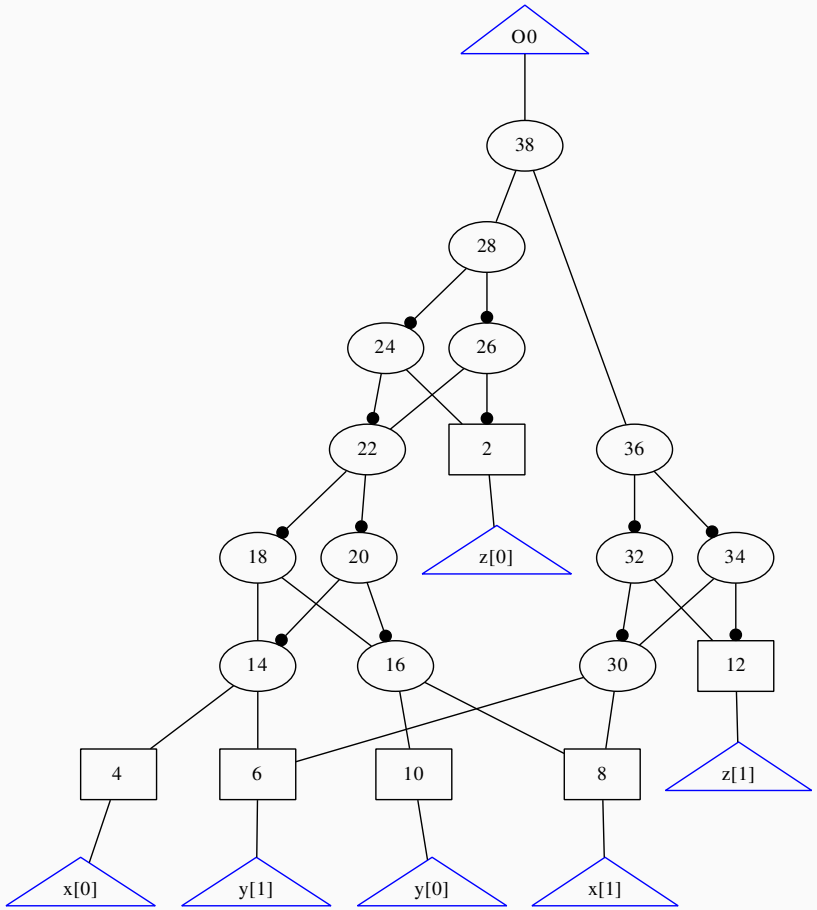
- current state-of-the-art for **quantifier-free** bit-vector formulas
- rewriting + simplifications + eager reduction to SAT

- efficient in practice
- may suffer from an exponential blow-up in the formula size

- may not scale well for increasing bit-widths

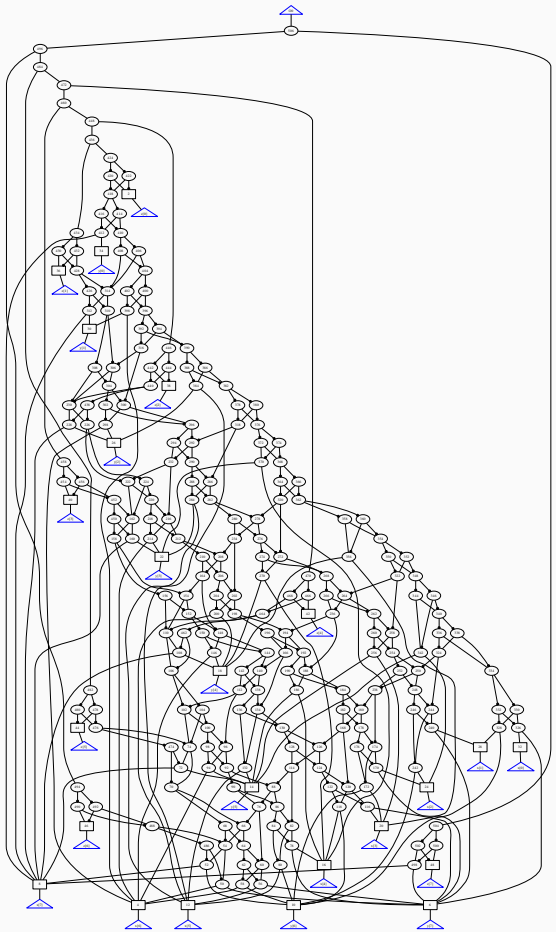
Bit-Blasting

Example $x_{[2]} * y_{[2]} = z_{[2]}$



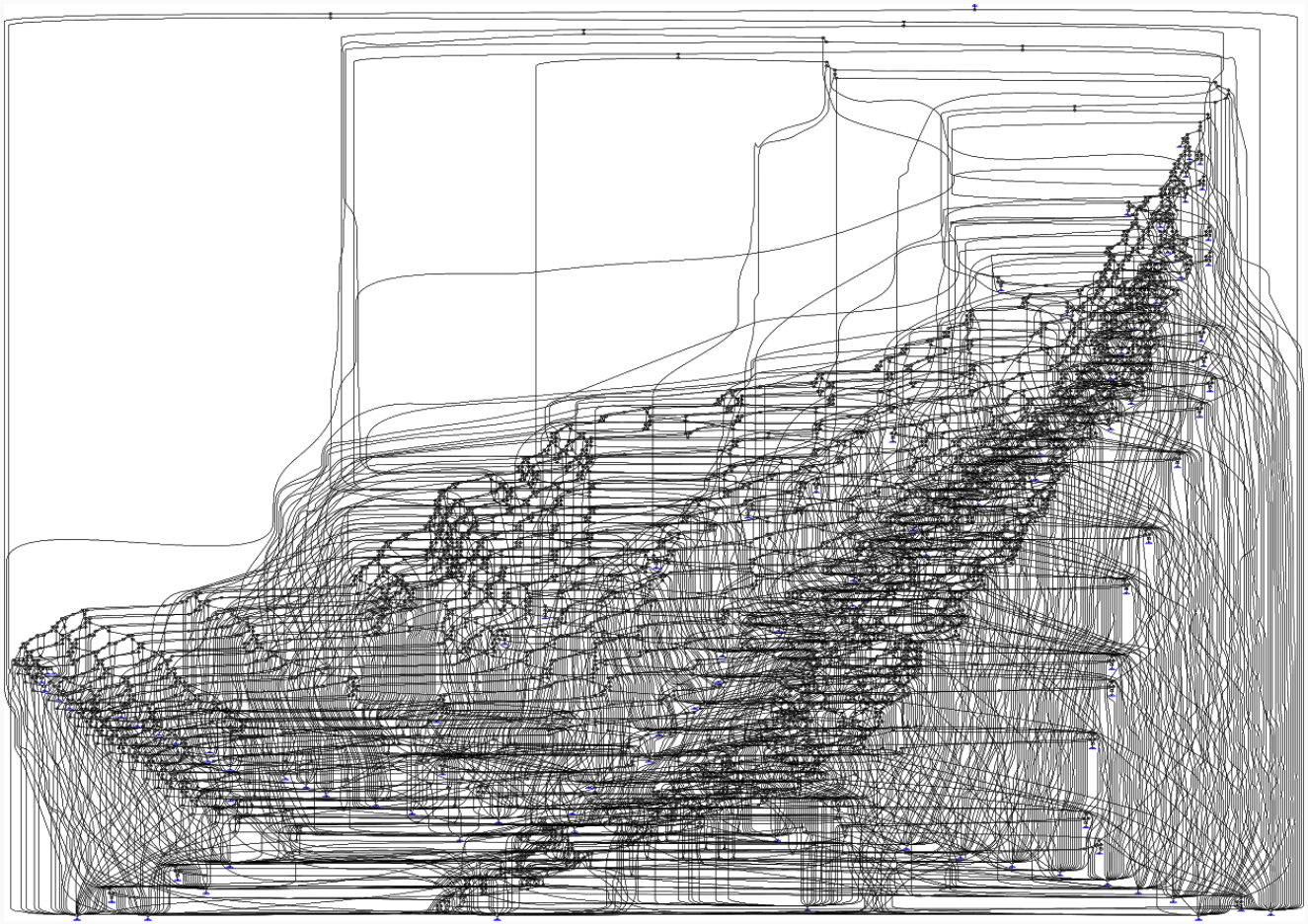
Bit-Blasting

Example $x_{[8]} * y_{[8]} = z_{[8]}$

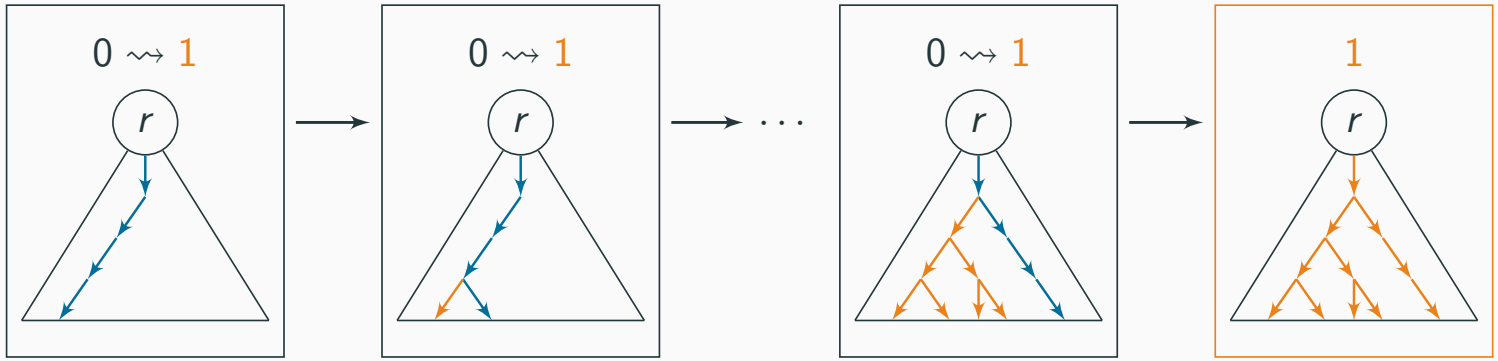


Bit-Blasting

Example $x_{[32]} * y_{[32]} = z_{[32]}$



Ternary Propagation-Based Local Search



- assume satisfiability, start with **initial assignment**
- **propagate** target values towards inputs
- iteratively improve current state until **solution** is found

- **orthogonal approach**
- lifts concept of **backtracing** from ATPG to the **word-level**
- **without** bit-blasting, **no** SAT solver

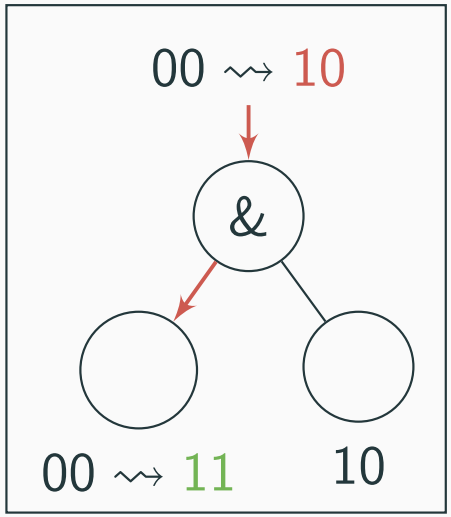
- **not able** to determine **unsatisfiability**
- Probabilistically Approximately Complete (PAC) [Hoos, AAAI'99]
  - guaranteed to find a solution if there is one

Ternary Propagation-Based Local Search

- **non-deterministic algorithm**
- two main sources of **non-determinism**:
  - **propagation path** selection
    - multiple possible paths
  - **propagation value** selection
    - multiple possible values
- **down-propagation** of target values with respect to **constant bits**

Propagation Value Selection

without changing other inputs



inverse value

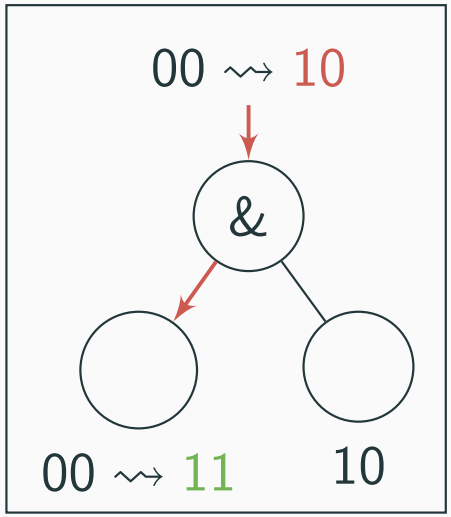
after changing other inputs

consistent value

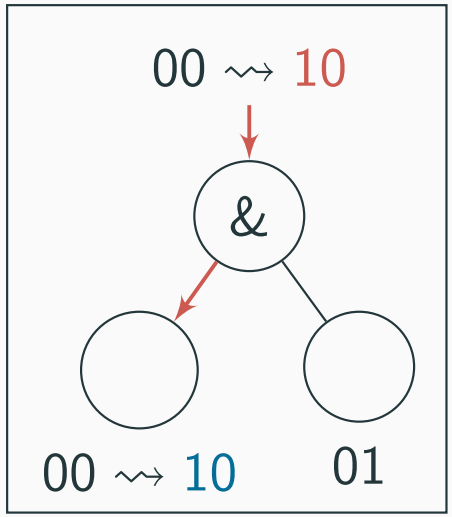
wrt const bits

Propagation Value Selection

without changing other inputs



inverse value

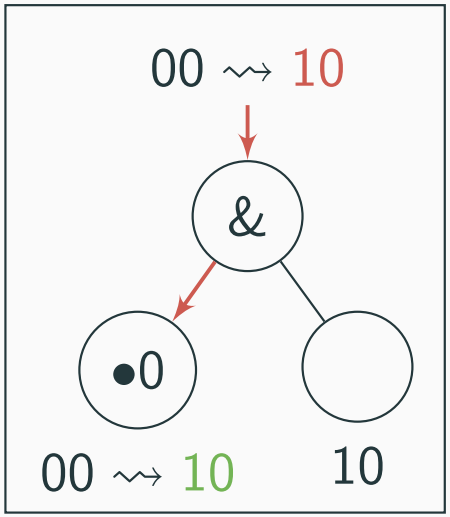


consistent value

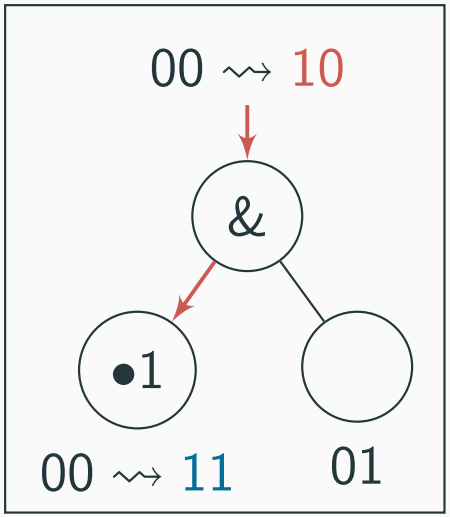
after changing other inputs

Propagation Value Selection

without changing other inputs



inverse value
wrt const bits

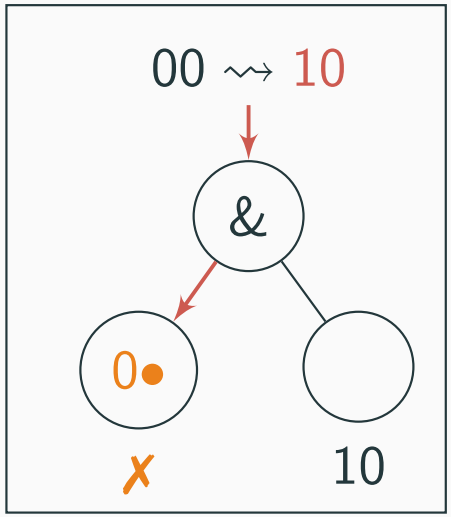


consistent value
wrt const bits

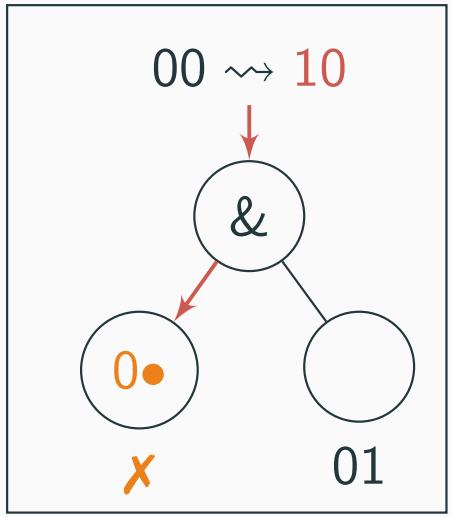
after changing other inputs

Propagation Value Selection

without changing other inputs



inverse value
wrt const bits

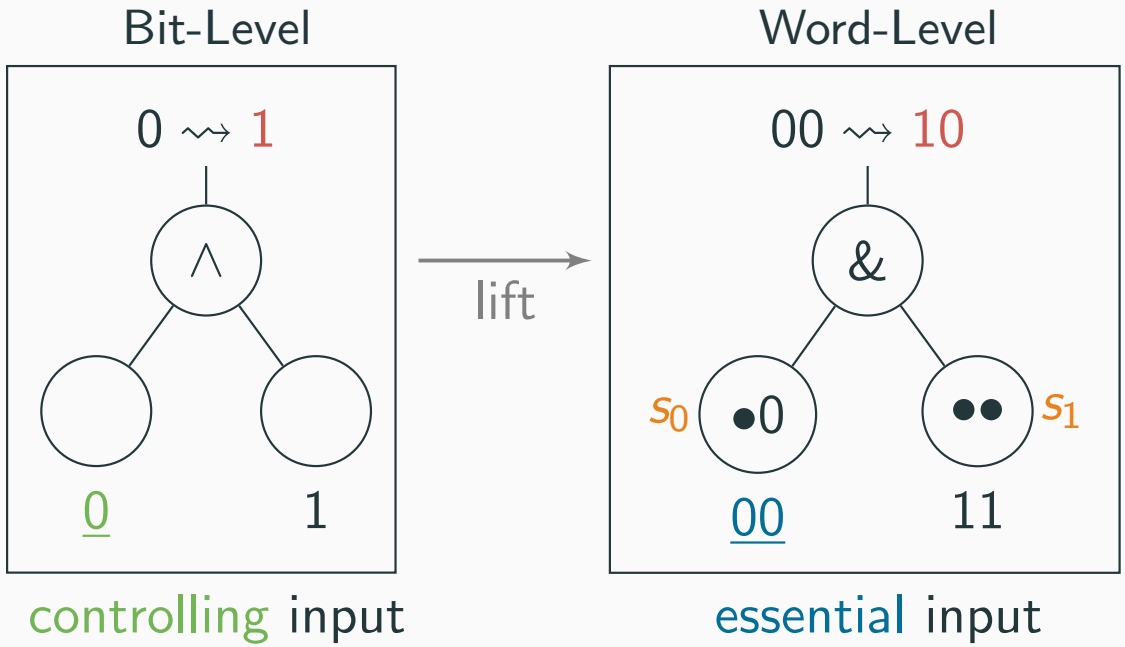


consistent value
wrt const bits

after changing other inputs

- inverse and consistent value **not always possible**
- **symbolic** invertibility/consistency conditions
- **break and restart** propagation if **consistency** condition **false**

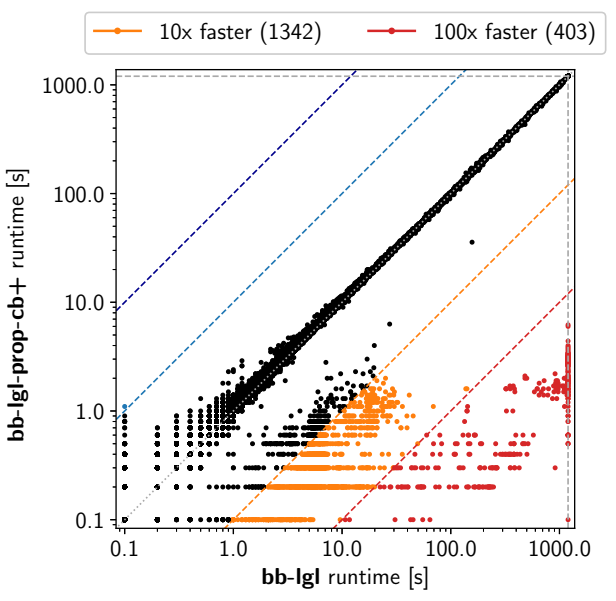
Propagation Path Selection



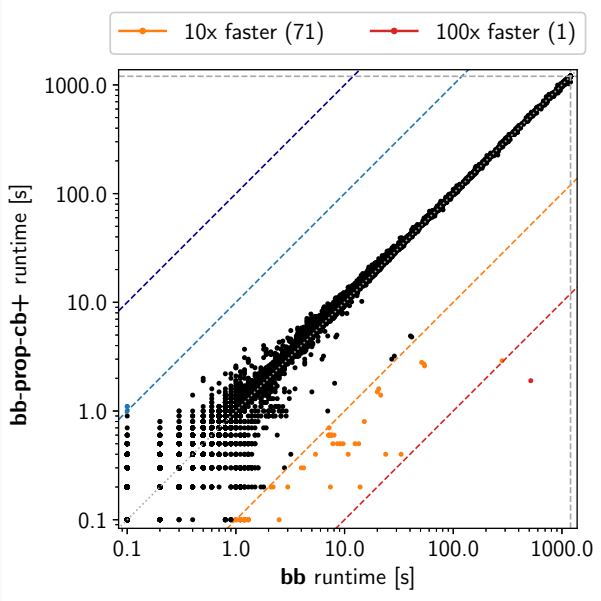
- s_0 is essential if there **does not** exist an inverse value for s_1
- s_1 is essential if there **does not** exist an inverse value for s_0

Results

- implemented in our **new** SMT solver **Bitwuzla**



Lingeling SAT back end



CaDiCaL SAT back end

- **sequential portfolio** (first run LS, then fall back to bit-blasting)
- **all** 41,713 benchmarks in SMT-LIB QF_BV
- winner of division QF_BV in the SMT-COMP 2020

Conclusion

- SMT solvers enable us to exploit the **structure of a problem**
- at the **backend** for many applications in AHA
- performance and scalability **key requirement**

- **continuous effort** to **improve performance**
  for problems and applications in AHA

References


H. H. Hoos. *On the Run-time Behaviour of Stochastic Local Search Algorithms for SAT.* In Proc. of AAAI/IAAI'99, pages 661–666, AAAI Press / The MIT Press, 1999.