More on the Complexity of Quantifier-Free Fixed-Size Bit-Vector Logics

Andreas Fröhlich, Gergely Kovásznai, Armin Biere

Institute for Formal Models and Verification
Johannes Kepler University, Linz, Austria
http://fmv.jku.at

CSR 2013
June 25 - June 29, 2013
Ekaterinburg, Russia

JOHANNES KEPLER UNIVERSITY LINZ | JKU

Motivation

- How does the *encoding* of the bit-widths affect the *complexity* of satisfiability checking for BV logics?
- In practice *logarithmic* (e.g. binary, decimal, hexadecimal) encoding is used (in contrast with unary encoding)

Example in SMT2

```
(set-logic QF_BV)
(declare-fun x () (_ BitVec 1000000))
(declare-fun y () (_ BitVec 1000000))
(declare-fun z () (_ BitVec 1000000))
(assert (= z (bvadd x y)))
(assert (= z (bvshl x (_ bv1 1000000))))
(assert (distinct x y))
```

Using Boolector:

- input file of 225 bytes in SMT2 format
- 129 MB in AIGER format; 843 MB in DIMACS format

Motivation

- How does the *encoding* of the bit-widths affect the *complexity* of satisfiability checking for BV logics?
- In practice *logarithmic* (e.g. binary, decimal, hexadecimal) encoding is used (in contrast with unary encoding)

Example in SMT2

```
(set-logic QF_BV)
(declare-fun x () (_ BitVec 1000000))
(declare-fun y () (_ BitVec 1000000))
(declare-fun z () (_ BitVec 1000000))
(assert (= z (bvadd x y)))
(assert (= z (bvshl x (_ bv1 1000000))))
(assert (distinct x y))
```

Using Boolector:

- input file of 225 bytes in SMT2 format
- 129 MB in AIGER format; 843 MB in DIMACS format

Previous Work

Let $\mathrm{QF\_BV}$ be the set of bit-vector formulas with *binary* encoding and all common bit-vector operations, e.g. bitwise operations, arithmetic operations, concatenation, slicing, shifts, relational operations, ...

- $\mathrm{QF\_BV}$ is $\mathrm{NE{\scriptsize XP}T{\scriptsize IME}}$-complete
- Proof:
  - $\mathrm{QF\_BV}$ is $\mathrm{NE{\scriptsize XP}T{\scriptsize IME}}$-hard:

$$\mathrm{DQBF} \xrightarrow{\text{polynomially}} \mathrm{QF\_BV}$$

  - $\mathrm{QF\_BV} \in \mathrm{NE{\scriptsize XP}T{\scriptsize IME}}$:

$$\mathrm{QF\_BV} \xrightarrow{\text{exponentially}} \mathrm{SAT} \in \mathrm{NP}$$

Completeness Results

How does restricting the set of operations affect the complexity?

- $\mathrm{QF\_BV}_{\ll c}$: bitwise operations, equality, and shift by any constant
  → $\mathrm{QF\_BV}_{\ll c}$ is NEXPTIME-complete
- $\mathrm{QF\_BV}_{\ll 1}$: bitwise operations, equality, and shift by only 1
  → $\mathrm{QF\_BV}_{\ll 1}$ is PSPACE-complete
- $\mathrm{QF\_BV}_{bw}$: bitwise operations and equality
  → $\mathrm{QF\_BV}_{bw}$ is NP-complete

Completeness Results

How does restricting the set of operations affect the complexity?

- $\mathrm{QF\_BV}_{\ll c}$: bitwise operations, equality, and shift by any constant
  $\rightarrow \mathrm{QF\_BV}_{\ll c}$ is NExpTime-complete
- $\mathrm{QF\_BV}_{\ll 1}$: bitwise operations, equality, and shift by only 1
  $\rightarrow \mathrm{QF\_BV}_{\ll 1}$ is PSpace-complete
- $\mathrm{QF\_BV}_{bw}$: bitwise operations and equality
  $\rightarrow \mathrm{QF\_BV}_{bw}$ is NP-complete

Completeness Results

How does restricting the set of operations affect the complexity?

- $\mathrm{QF\_BV}_{\ll c}$: bitwise operations, equality, and shift by any constant
  $\rightarrow$ $\mathrm{QF\_BV}_{\ll c}$ is NEXPTIME-complete
- $\mathrm{QF\_BV}_{\ll 1}$: bitwise operations, equality, and shift by only 1
  $\rightarrow$ $\mathrm{QF\_BV}_{\ll 1}$ is PSPACE-complete
- $\mathrm{QF\_BV}_{bw}$: bitwise operations and equality
  $\rightarrow$ $\mathrm{QF\_BV}_{bw}$ is NP-complete

Completeness Results

How does restricting the set of operations affect the complexity?

- $\mathrm{QF\_BV}_{\ll c}$: bitwise operations, equality, and shift by any constant
  $\rightarrow$ $\mathrm{QF\_BV}_{\ll c}$ is NExpTime-complete

- $\mathrm{QF\_BV}_{\ll 1}$: bitwise operations, equality, and shift by only 1
  $\rightarrow$ $\mathrm{QF\_BV}_{\ll 1}$ is PSpace-complete

- $\mathrm{QF\_BV}_{bw}$: bitwise operations and equality
  $\rightarrow$ $\mathrm{QF\_BV}_{bw}$ is NP-complete

Complexity: $\mathrm{QF\_BV}_{\ll 1}$

$\mathrm{QF\_BV}_{\ll 1}$ is PSPACE-complete:

- $\mathrm{QF\_BV}$ is PSPACE-hard:

$$\mathrm{QBF} \xrightarrow{\text{polynomially}} \mathrm{QF\_BV}_{\ll 1}$$

- $\mathrm{QF\_BV} \in$ PSPACE:

$$\mathrm{QF\_BV}_{\ll 1} \xrightarrow{\text{polynomially}} \text{Sequential Circuits}$$

Complexity: $bvf_{\ll 1}$ is PSPACE-hard

Quantified Boolean Formulas (QBF):

- Variable dependencies are *implicitely* specified by prefix order
- Dependencies represent a total order

Example DQBF

$$\begin{aligned}\forall u_2 \exists x \forall u_1 \exists y \forall u_0 \exists z \,.\, F \;&=\\ \forall u_2 \exists x \forall u_1 \exists y \forall u_0 \exists z \,.\, &(x \vee y \vee \neg u_2 \vee \neg u_1) \wedge\\ &(x \vee \neg z \vee u_2 \vee \neg u_1 \vee \neg u_0) \wedge\\ &(\neg y \vee z \vee \neg u_1 \vee u_0) \wedge\\ &(\neg x \vee y \vee \neg u_2) \wedge\\ &(\neg x \vee \neg z \vee u_2 \vee \neg u_0)\end{aligned}$$

- QBF is PSPACE-complete

Complexity: $bvf_{\ll 1}$ is PSPACE-hard

$$\Big((X^{[8]} \mid Y^{[8]} \mid{\sim} U_2^{[8]} \mid{\sim} U_1^{[8]}) \;\&\; (X \mid{\sim} Z^{[8]} \mid U_2 \mid{\sim} U_1 \mid{\sim} U_0^{[8]}) \;\&$$
$$({\sim} Y \mid Z \mid{\sim} U_1 \mid U_0) \;\&\; ({\sim} X \mid Y \mid{\sim} U_2) \;\&$$
$$({\sim} X \mid{\sim} Z \mid U_2 \mid{\sim} U_0) \Big) \;=\; {\sim} 0^{[8]}$$

1. Eliminate the quantifier prefix
2. Replace logical connectives with bitwise operators
3. Replace Boolean variables with bit-vector variables of *bit-width* 2^k

   k: number of universal variables in the QBF

So far, corresponds to $\exists u_2, u_1, u_0, x, y, z \;.\; F$

Complexity: $bvf_{\ll 1}$ is PSPACE-hard

$$\Big((X^{[8]} \mid Y^{[8]} \mid\!\sim U_2^{[8]} \mid\!\sim U_1^{[8]})\ \&\ (X \mid\!\sim Z^{[8]} \mid U_2 \mid\!\sim U_1 \mid\!\sim U_0^{[8]})\ \&$$
$$(\sim Y \mid Z \mid\!\sim U_1 \mid U_0)\ \&\ (\sim X \mid Y \mid\!\sim U_2)\ \&$$
$$(\sim X \mid\!\sim Z \mid U_2 \mid\!\sim U_0)\Big) \;=\; \sim 0^{[8]}$$

4. *Universal* vars ← Assign *binary magic numbers* to U_is!

$$U_2 := \begin{bmatrix} 0\\0\\0\\0\\1\\1\\1\\1 \end{bmatrix}, U_1 := \begin{bmatrix} 0\\0\\1\\1\\0\\0\\1\\1 \end{bmatrix}, U_0 := \begin{bmatrix} 0\\1\\0\\1\\0\\1\\0\\1 \end{bmatrix}$$

Complexity: $bvf_{\ll 1}$ is PSPACE-hard

$$\Big((X^{[8]} \mid Y^{[8]} \mid\sim U_2^{[8]} \mid\sim U_1^{[8]}) \,\&\, (X \mid\sim Z^{[8]} \mid U_2 \mid\sim U_1 \mid\sim U_0^{[8]}) \,\&$$
$$(\sim Y \mid Z \mid\sim U_1 \mid U_0) \,\&\, (\sim X \mid Y \mid\sim U_2) \,\&$$
$$(\sim X \mid\sim Z \mid U_2 \mid\sim U_0)\Big) \;=\; \sim 0^{[8]}$$

4. *Universal* vars ← Assign *binary magic numbers* to U_is!

$$U_2 := \begin{bmatrix} 0\\0\\0\\0\\1\\1\\1\\1 \end{bmatrix}, U_1 := \begin{bmatrix} 0\\0\\1\\1\\0\\0\\1\\1 \end{bmatrix}, U_0 := \begin{bmatrix} 0\\1\\0\\1\\0\\1\\0\\1 \end{bmatrix}$$

Complexity: $bvf_{\ll 1}$ is PSPACE-hard

$$\Big((X^{[8]} \mid Y^{[8]} \mid\!\sim U_2^{[8]} \mid\!\sim U_1^{[8]}) \;\&\; (X \mid\!\sim Z^{[8]} \mid U_2 \mid\!\sim U_1 \mid\!\sim U_0^{[8]}) \;\&$$

$$(\sim Y \mid Z \mid\!\sim U_1 \mid U_0) \;\&\; (\sim X \mid Y \mid\!\sim U_2) \;\&$$

$$(\sim X \mid\!\sim Z \mid U_2 \mid\!\sim U_0)\Big) \;=\; \sim 0^{[8]}$$

4. *Universal* vars ← Assign *binary magic numbers* to U_is!

For $m \in \{0, 1, 2\}$, add:

$$T_m^{[8]} = \left(\bigwedge_{0 \le i < m} U_i^{[8]}\right) \oplus U_m^{[8]}$$

$$T_m^{[8]} = U_m^{[8]} \ll 1^{[8]}$$

Complexity: $bvf_{\ll 1}$ is PSPACE-hard

$$\Big((X^{[8]} \mid Y^{[8]} \mid\!\sim U_2^{[8]} \mid\!\sim U_1^{[8]}) \ \&\ (X \mid\!\sim Z^{[8]} \mid U_2 \mid\!\sim U_1 \mid\!\sim U_0^{[8]}) \ \&$$
$$(\sim Y \mid Z \mid\!\sim U_1 \mid U_0) \ \&\ (\sim X \mid Y \mid\!\sim U_2) \ \&$$
$$(\sim X \mid\!\sim Z \mid U_2 \mid\!\sim U_0) \Big) \ = \ \sim 0^{[8]}$$

$$\wedge \ \boxed{\bigwedge_{m \in \{0,1,2\}} \left(\left(\bigwedge_{0 \leq i < m} U_i \right) \oplus U_m = U_m \ll 1^{[8]} \right)}$$

So far, corresponds to $\forall u_2, u_1, u_0 \exists x, y, z \ . \ F$

Complexity: $bvf_{\ll 1}$ is PSPACE-hard

$$\Big((X^{[8]} \mid Y^{[8]} \mid\sim U_2^{[8]} \mid\sim U_1^{[8]}) \;\&\; (X \mid\sim Z^{[8]} \mid U_2 \mid\sim U_1 \mid\sim U_0^{[8]}) \;\&\;$$
$$(\sim Y \mid Z \mid\sim U_1 \mid U_0) \;\&\; (\sim X \mid Y \mid\sim U_2) \;\&\;$$
$$(\sim X \mid\sim Z \mid U_2 \mid\sim U_0)\Big) \;=\; \sim 0^{[8]}$$
$$\wedge \bigwedge_{m \in \{0,1,2\}} \left(\left(\bigwedge_{0 \le i < m} U_i \right) \oplus U_m = U_m \ll 1^{[8]} \right)$$

5 *Existential* vars $\leftarrow$ Represent *Skolem-functions* as bit-vectors!

$U_1 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$ is the innermost variable $Y = \begin{bmatrix} y_7 \\ y_6 \\ y_5 \\ y_4 \\ y_3 \\ y_2 \\ y_1 \\ y_0 \end{bmatrix}$ depends on.

Complexity: $bvf_{\ll 1}$ is PSPACE-hard

$$\Big((X^{[8]} \mid Y^{[8]} \mid\sim U_2^{[8]} \mid\sim U_1^{[8]}) \,\&\, (X \mid\sim Z^{[8]} \mid U_2 \mid\sim U_1 \mid\sim U_0^{[8]}) \,\&$$
$$(\sim Y \mid Z \mid\sim U_1 \mid U_0) \,\&\, (\sim X \mid Y \mid\sim U_2) \,\&$$
$$(\sim X \mid\sim Z \mid U_2 \mid\sim U_0)\Big) \;=\; \sim 0^{[8]}$$
$$\wedge \bigwedge_{m \in \{0,1,2\}} \left(\left(\bigwedge_{0 \le i < m} U_i\right) \oplus U_m = U_m \ll 1^{[8]}\right)$$

5 *Existential* vars ← Represent *Skolem-functions* as bit-vectors!

Let U_m be the innermost universal variable an existential variable E depends on. For $m > 0$, add:

$$U'_m = \sim ((U_m \ll 1) \oplus U_m)$$

$$(E \,\&\, U'_m) = ((E \ll 1) \,\&\, U'_m)$$

Complexity: $bvf_{\ll 1}$ is PSPACE-hard

$$\Big((X^{[8]} \mid Y^{[8]} \mid\!\sim U_2^{[8]} \mid\!\sim U_1^{[8]}) \;\&\; (X \mid\!\sim Z^{[8]} \mid U_2 \mid\!\sim U_1 \mid\!\sim U_0^{[8]}) \;\&$$

$$(\sim Y \mid Z \mid\!\sim U_1 \mid U_0) \;\&\; (\sim X \mid Y \mid\!\sim U_2) \;\&$$

$$(\sim X \mid\!\sim Z \mid U_2 \mid\!\sim U_0)\Big) \;=\; \sim 0^{[8]}$$

$$\wedge \bigwedge_{m \in \{0,1,2\}} \left(\left(\bigwedge_{0 \le i < m} U_i\right) \oplus U_m = U_m \ll 1^{[8]}\right)$$

$$\wedge\; U_2' = \sim((U_2 \ll 1) \oplus U_2) \wedge (X \;\&\; U_2') = ((X \ll 1) \;\&\; U_2')$$

$$\wedge\; U_1' = \sim((U_1 \ll 1) \oplus U_1) \wedge (Y \;\&\; U_1') = ((Y \ll 1) \;\&\; U_1')$$

Corresponds to $\forall u_2 \exists x \forall u_1 \exists y \forall u_0 \exists z \;.\; F$

Complexity: $\mathrm{QF\_BV}_{\ll 1} \in \mathrm{PSPACE}$

- Existing work for non-fixed-size bit-vectors resp. quantifier-free Presburger arithmetic with bitwise operations ($\mathrm{QFPA{\small BIT}}$):

  $\mathrm{QFPA{\small BIT}} \xrightarrow{\text{polynomially}}$ Sequential Circuits

  A. Spielmann, V. Kuncak, *Synthesis for Unbounded Bit-Vector Arithmetic.* In: Proc. IJCAR'12.

- A flat normal form of the original formula is created:

  Logical combination of certain atomic expressions.

- For each atomic expression a direct translation into an atomic sequential circuit can be given.
  - The result is the logical combination of the atomic circuits.

- Can be adopted for fixed-size bit-vectors of bit-width 2^n by introducing a n-bit counter.

  Counter can be realized using bitwise operations and shift by 1.

Complexity: $\mathrm{QF\_BV}_{bw} \in \mathrm{NP}$

- Existing work on bit-width reduction of bit-vector formulas.

  P. Johannsen, *Reducing Bitvector Satisfiability Problems to Scale Down Design Sizes for RTL Property Checking.* In: Proc. HLDVT'01.

- Basically: "It is enough to consider as many bits as there are equalities in the formula."

  Different bit-positions do not interact with each other and a witness for every falsified equality can be given.
  $\rightarrow$ A reduction to a *bit-width bounded* set of formulas exists.

- If $S \subset \mathrm{QF\_BV}$ is bit-width bounded, $S \in \mathrm{NP}$.

  G. Kovásznai, A. Fröhlich, A. Biere, *On the Complexity of Fixed-Size Bit-Vector Logics with Binary Encoded Bit-Width.* In: Proc. SMT'12.

Practical Considerations

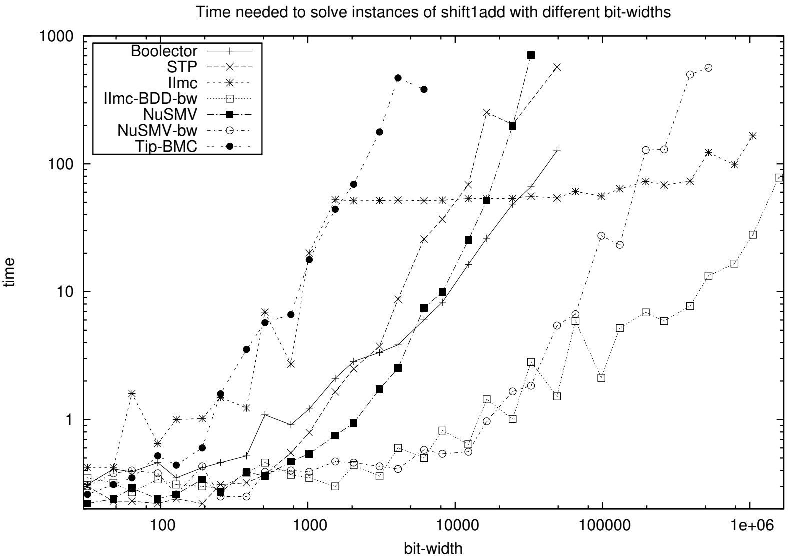
- PSPACE-inclusion also holds for addition, indexing, multiplication by constant, relational operations, ...
- Bit-blasting can cause exponential growth.
- Use Model Checkers to solve $\mathrm{QF\_BV}_{\ll 1}$ formulas more efficiently.

  A. Fröhlich, G. Kovásznai, A. Biere, *Efficiently Solving Bit-Vector Problems Using Model Checkers* In: Proc. SMT'13 (to appear).

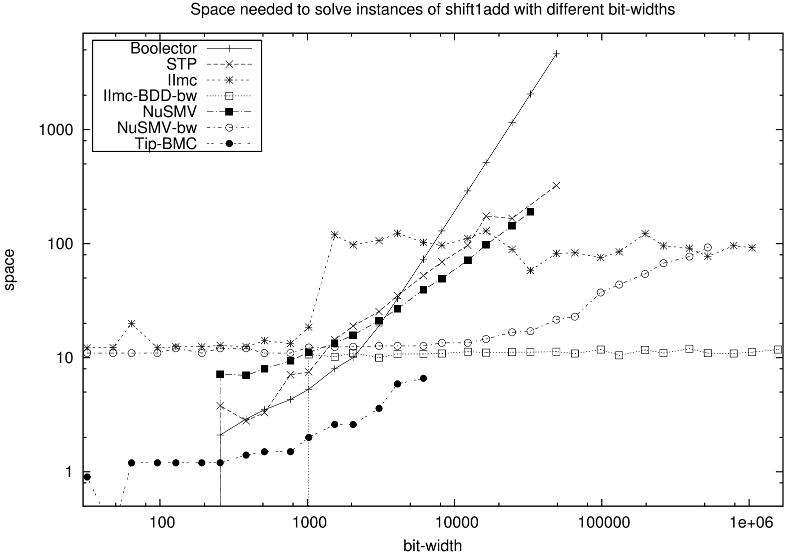
Example in SMT2

```
(set-logic QF_BV)
(declare-fun x () (_ BitVec 1000000))
(declare-fun y () (_ BitVec 1000000))
(declare-fun z () (_ BitVec 1000000))
(assert (= z (bvadd x y)))
(assert (= z (bvshl x (_ bv1 1000000))))
(assert (distinct x y))
```

Experimental Results



Experimental Results



Conclusion

Theoretical Results:

- $\mathrm{QF\_BV}_{\ll_c}$ is $\mathrm{NExpTime}$-complete.
- $\mathrm{QF\_BV}_{\ll 1}$ is PSpace-complete.
- $\mathrm{QF\_BV}_{bw}$ is NP-complete.

Future Work:

- Is Presburger arithmetic on fixed-size bit-vectors still NP-complete?
- Can we use our approach to solve industrial benchmarks more efficiently?
- How can state-of-the-art SMT solvers profit from techniques used in model checkers?