# A Theory of Finite Sets, Lists, and Maps for the SMT-LIB Standard

Daniel Kroening Philipp Rümmer Georg Weissenbacher

Oxford University Computing Laboratory

philr@comlab.ox.ac.uk

SMT Workshop 2009 on CADE 22

2 August 2009

# Outline

- Motivation of new theories for SMT-LIB 2
- Proposal of theories
- Some examples (VDM, Event-B)
- SMT-LIB 2 converter
- Practical and theoretical issues

More information + implementation:
`http://www.cprover.org/SMT-LIB-LSM/`

# Motivation: SMT Applications

## Bounded Model Checking for C, C++ (CBMC)

- Until recently: based on SAT solving
- SMT backend using bit-vectors + arrays (QF_AUFBV)
- Planned: other theories to handle library models (e.g., STL), memory models, etc.

## Model-based test-case generation

- Modelling languages like UML/OCL (e.g, state charts), Simulink/Stateflow, Lustre
- E.g., by bounded model checking, constraint solving
- EU projects Mogentes, CESAR

# Motivation: SMT Applications (2)

## Analysis of requirements + architecture specifications

- Consistency, coverage, animation, etc.
- Languages developed in EU project CESAR

## System development in Event-B, VDM

- Set-theoretic specification languages
- Proof obligations for invariant preservation, refinement, etc.
- Event-B tool with built-in proof assistant (Rodin)
  $\Rightarrow$ Currently no usage of SMT

# SMT-LIB is Great!

- Simple format
- Many available solvers, backend easily changeable
- Experiences in CBMC:
  Significantly reduced implementation effort
  (Compared to native SMT solver formats)

# SMT-LIB is Great! *but there are some issues . . .*

## Issues with SMT-LIB 1.2, from our point of view

- Fixed types of arrays indices/values
  (e.g., no boolean arrays)
- Separation between formulae and terms
- No constant arrays
- *. . . missing datatypes*

$\Rightarrow$ Many problems are fixed by upcoming SMT-LIB 2

# Solutions to Missing Datatypes

- Much can be encoded in
  arrays + uninterpreted functions + axioms
  $\Rightarrow$ Difficult to build decision procedures
  (for decidable fragments)
- Many solvers offer extensions + further theories
  (e.g., algebraic datatypes)
  $\Rightarrow$ Not standardised, against the SMT-LIB idea
- *Introduce further SMT-LIB theories*

# Proposal for Additional SMT-LIB 2 Theories

## Datatypes inspired by VDM-SL

- Tuples
- Lists
- (Finite) Sets
- (Finite) Partial Maps

- Defined as parametric SMT-LIB 2 theories
- Semantics in terms of classical set theory

# Signature of the Datatypes

| **Tuples** | **Sets** | **Lists** | **Maps** |
|---|---|---|---|
| (Tuple$_n$ T$_1$ ... T$_n$) | (Set T) | (List T) | (Map S T) |
| tuple $(x_1, \ldots, x_n)$ | emptySet $\emptyset$ | nil [] | emptyMap $\emptyset$ |
| project $x_k$ | insert $M \cup \{x\}$ | cons $x :: L$ | apply $f(x)$ |
| product $M_1 \times \cdots \times M_n$ | in $\in$ | head | overwrite $\lhd\!\!+$ |
| | subset $\subseteq$ | tail | domain |
| | union $\cup$ | append $\curvearrowright$ | range |
| | inter $\cap$ | length $\lvert l \rvert$ | restrict $\lhd$ |
| | setminus $\setminus$ | nth $l_k$ | subtract $\lhd\!\!\!-$ |
| | card $\lvert M \rvert$ | inds $\{1, \ldots, \lvert l \rvert\}$ | |
| | | elems $\{l_1, \ldots, l_{\lvert l \rvert}\}$ | |

# Defined Theories

- Sets with cardinality
- Sets + Tuples
- Lists with length
- Finite Maps
- Combined theories

# Defined Theories (preliminary decidability results)

- Sets with cardinality: non-nested: decidable
  nested + quantifiers: undecidable
  nested, quantifier-free: ???
- Sets + Tuples: undecidable
- Lists with length: word equations with equal-length predicate, known open problem
- Finite Maps: ???
- Combined theories: undecidable

# Example: Verification Cond. Generated by VDMTools

In VDM-SL notation:

$$\forall l : \mathbb{L}(\mathbb{Z}), i : \mathbb{N}.\ \big(i \in \mathsf{inds}(l) \Rightarrow \forall j \in \mathsf{inds}(l) \setminus \{i\}.\ j \in \mathsf{inds}(l)\big)$$

In SMT-LIB notation:

```
(forall ((l (List Int)) (i Int))
  (implies
     (and (>= i 0) (in i (inds l)))
     (forall (j Int)
       (implies
          (in j (setminus (inds l) (set i)))
          (in j (inds l))))))
```

# Event-B File System Case Study (delete/inv8)

$$
\begin{array}{rl}
 & parent \in objects \setminus \{root\} \rightarrow objects, \\
 & obj \in objects \setminus \{root\}, \quad des \subseteq objects, \\
 & des = (tcl(parent)) \sim [\{obj\}], \quad objs = des \cup \{obj\} \\
\Rightarrow & objs \lhd parent \in (objects \setminus objs) \setminus \{root\} \rightarrow objects \setminus objs
\end{array}
$$

```
:extrafuns((objects, des, objs (Set OBJECT))
           (parent (Map OBJECT OBJECT))
           (obj OBJECT))

(implies ... (and
  (= (domain (subtract parent objs))
     (setminus objects
               objs (insert emptySet root)))
  (subset (range (subtract parent objs))
          (setminus objects objs))
))
```

# Status of the Proposal

- Syntax + Semantics of theories is defined
  $\Rightarrow$ In collaboration with Cesare Tinelli
- Parser + type checker available
- Meaningful sublogics still to be identified
- Decidability is being investigated

# SMT-LIB 2 Front-end

$\Rightarrow$ *Finished just in time for workshop*

- Motivation: ease the adoption of SMT-LIB 2 + theories
- Parser + type checker for (almost) complete SMT-LIB 2
- All proposed theories are supported

- Current backend: converter to SMT-LIB 1
  $\Rightarrow$ We consider further backends (like: native Z3)

- *Of course: we would prefer direct support by SMT solvers*

`http://www.cprover.org/SMT-LIB-LSM/`

# Current (Incomplete) Axiomatisation of Theories

- Sets with cardinality: arrays + uninterpreted functions + axioms
- Tuples, lists, maps: uninterpreted functions + axioms

# Experiences with SMT-LIB 2 in CBMC

- CBMC already has an SMT-LIB 2 backend
- Significantly reduced implementation effort
  Code size $\approx$ 50% compared to SMT-LIB 1
- Currently: Tuples are used, other theories in near future

## Most positive changes compared to SMT-LIB 1.2

- No term/formula distinction
- Possibility of polymorphic arrays
- Let-expressions as terms

# ~~Fund~~ *Benchmark* Raising

- CBMC verification conditions
- Received $\approx$ 5500 verification conditions from Event-B community
- Still to be converted to SMT-LIB 2 format

# Conclusion

Trade-off when defining theories:

- Generality → good for users
- Implementation complexity → good for tool writers
- Decidability

⇒ We hope that we have found a compromise
⇒ Comments are welcome!

- We received very positive feedback from Event-B community

Thanks for your attention!

`http://www.cprover.org/SMT-LIB-LSM/`