Solving Relational Constraints with Extensions to a Theory of Finite Sets in SMT

Baoluo Meng, Andrew Reynolds, Cesare Tinelli
The University of Iowa

Our Research

- **Goal**: Solving Relational Constraints with Extensions to a Theory of Finite Sets in SMT
- Propose relational extensions to a theory of finite sets with cardinality
- Design and implement a calculus for the relational extensions in CVC4 SMT solver
- Develop a natural translator from Alloy to SMT

Related Work

- Alloy [1] is a declarative language for modeling and analyzing structurally-rich problems based on relational logic with built-in transitive closure and cardinality
  - The analysis of Alloy specification is performed automatically by the Alloy Analyzer – a SAT-based finite model finder

- To overcome the limitations of Alloy Analyzer, El Ghazi et al. translated Alloy kernel language to SMT-LIB language [3] and leveraged SMT solvers to solve resulting SMT formulas [6]
  - Limitation: Can only automatically disprove properties, but not prove them
  - Limitation: Limited support for numerical reasoning
- Extended the CVC4 native language to accept relational operators
  - E.g. CARD(TCLOSURE(S)) = 2 + CARD(S)

Motivation

- Support relational constructs and operators natively in SMT solvers
- Build a natural translator from Alloy to SMT-LIB
- Leverage SMT solvers to solve the resulting SMT formulas
  - Take advantage of other supported theories in SMT
  - Translation require much less quantifiers
- Can prove and disprove properties of Alloy specifications

Our Approach – Relational Extensions to a Theory of Finite Sets in SMT

- Kshitij Bansal et al. introduced a theory $T_F$ of finite sets in SMT [5]
- A parametric sort: Set(α) with sort α for set elements
- Constant and function symbols:
  - EMPTYSET : Set(α)
  - SINGLETON : α → Set(α)
  - UNION, INTERSECTION, DIFFERENCE : Set(α) × Set(α) → Set(α)
- Predicates symbols:
  - IS_IN : α × Set(α)
  - SUBSET : Set(α) × Set(α)
- A decision procedure for $T_F$ was implemented in CVC4 [4]
  - Also extended to support cardinality CARD : Set(α) → Int

Relational Extensions

- We propose extensions to $T_F$ with relational operators
  - Relational signature extensions:
    - TCLOSURE : Set(Tuple) → Set(Tuple)
    - TRANSPOSE : Set(Tuple) → Set(Tuple)
    - JOIN : Set(Tuple) × Set(Tuple) → Set(Tuple)
    - PRODUCT : Set(Tuple) × Set(Tuple) → Set(Tuple)
    - Tuple = (α₁, α₂, ..., αₙ) of arity n where αᵢ is a sort ∀ i ∈ [1,...,n]

- Developed a calculus for the relational extensions

A Relational Solver in CVC4

Future Work

- Identify decidable fragments of relational logic, for which our calculus is sound, complete and terminating
- Fully support relational reasoning with cardinality
  - E.g. CARD(TCLOSURE(S)) = 2 + CARD(S)
- Complete the implementation of translator from Alloy to SMT

Conclusion

- Fully support Alloy kernel language in SMT natively
- The initial version of the calculus for the extensions has been implemented in CVC4
- Extended the CVC4 native language to accept relational operators
- Modular, can solve constraints in combination with all other theories supported by CVC4

References