Third Mondex Workshop

University of York – October 5-6th, 2006

Mondex / Alloy

Last Updates

Tahina Ramananandro

École Normale Supérieure

Paris, France

Daniel Jackson

Massachusetts Institute of Technology

CSAIL Software Design

Cambridge MA, USA

Outline

- Work progress since May
- Improving the Model
- Using FOL theorem provers
- Conclusion and Future Work

What was done in May ?

- Z spec converted into Alloy modules
  - In a naive way
- All refinement theorems checked
  - But some constraint checks were missing

What was planned in May ?

- Improve formal model
  - More uniform treatment of existential theorems
  - Experiment with more Alloy-like idiom (eg, objects)
- Prove or argue small model theorem?
- Interface Alloy method with others

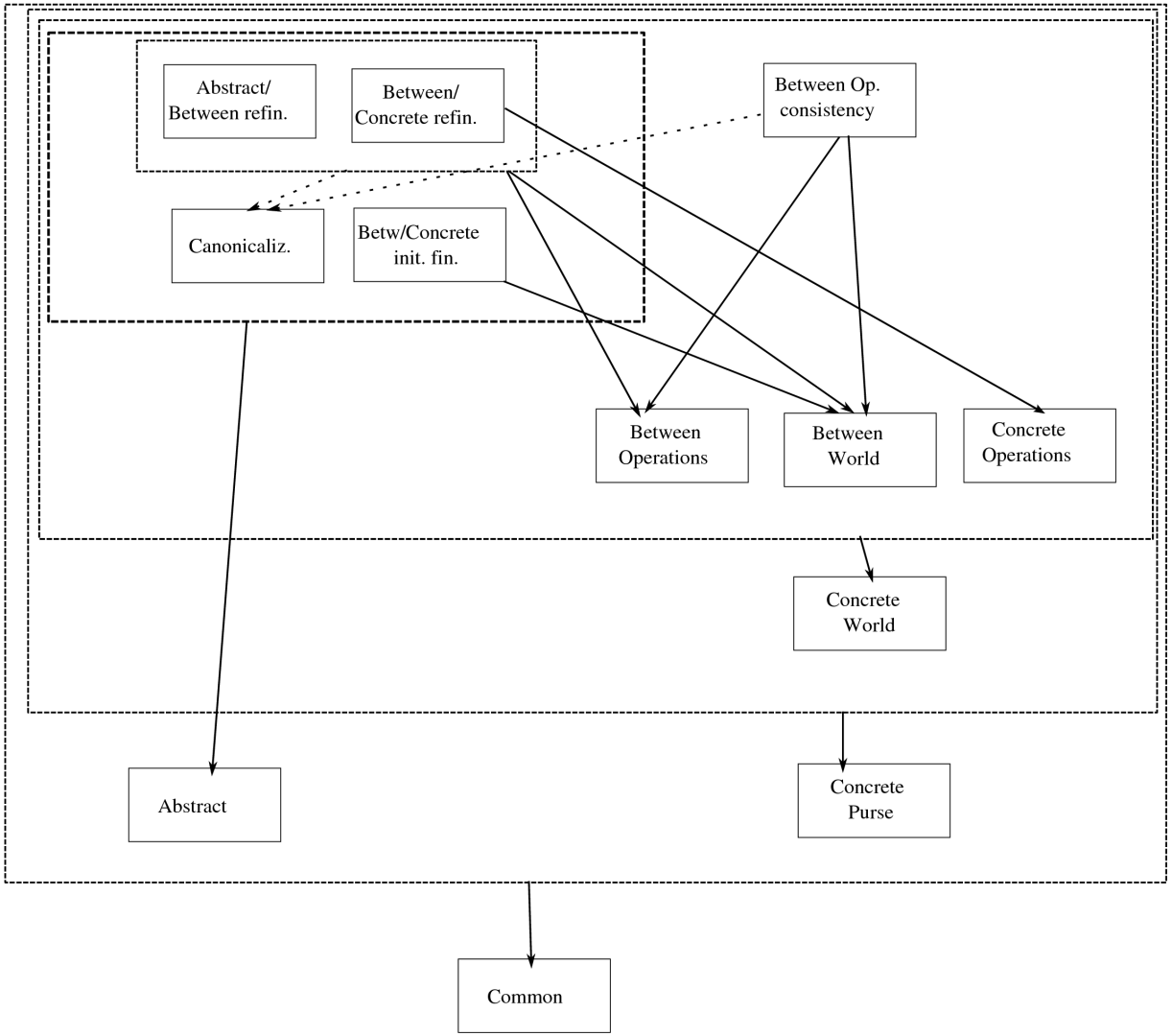
What has been done since May ?

- Improve formal model
  - More uniform, rigorous model
  - Weaker constraints
  - Constraints are no longer global, but integrated into theorems
  - However, no further bugs found
- Prove or argue small model theorem?
  - Mondex spec is FOL
    - if finiteness issues dropped
  - So, try to use FOL theorem provers
- Interface Alloy method with others
  - May be feasible (cf. future Alloy workshop)

Outline

- Work progress since May
- Improving the Model
- Using FOL theorem provers
- Conclusion and Future Work

Better modular organization



Coin sharing constraints

- Simulations showed that previous constraints were too strong
  - ```
    no p:ConPurse,pd:PayDetails {
     pd in p.exLog
     some pd.value & p.balance
    }
    ```
  - Prevents a purse from logging an aborted transaction with coins
- Newer constraints
  - Reason about the *maybeLost* and *definitelyLost* definitions
  - ```
    all c:ConWorld {
     no NAME.(c.conAuthPurse).balance
      & (maybeLost(c) + definitelyLost(c)).value
    }
    ```

Existential issue

- Can't guarantee object exists for every combination of field values
  - The empty model
  - To enforce existence with algebraic constraints would dramatically increase scope
- Solution :
  - Instead of ∃, construct *explicit witness* :
    `all c, c', a | some a' | P (c, c', a, a')`
    becomes
    ```
    all c, c', a |
    let a' = F(c, c', a) | P(c, c', a, a')
    ```
  - **Requires to get rid of global constraints**
    - **Integrate them into theorems**

Example : Between/Concrete

```
sig ConWorld {...}
pred Concrete (c:ConWorld) {...}
pred Between (b:ConWorld) {Concrete(b) and ...}

pred Rbc_constr (b,c:ConWorld, ...) {...}
pred Rbc (b,c:ConWorld) {...}

assert Rbc_Increase {
 all b,b',c,c':ConWorld, ... | {
  Concrete(c) and Concrete(c')
  Between(b)
  CIncrease(c,c',...)
  Rbc(b,c)
  Rbc_constr(b',c',...)
 } implies {
  Rbc(b',c')
  Increase(b,b',...)
 }
}
```

```
assert Increase_inv {
 all b,b':ConWorld,... | {
  Between(b)
  Increase(b,b',...)
 }
 implies Between(b')
}
```

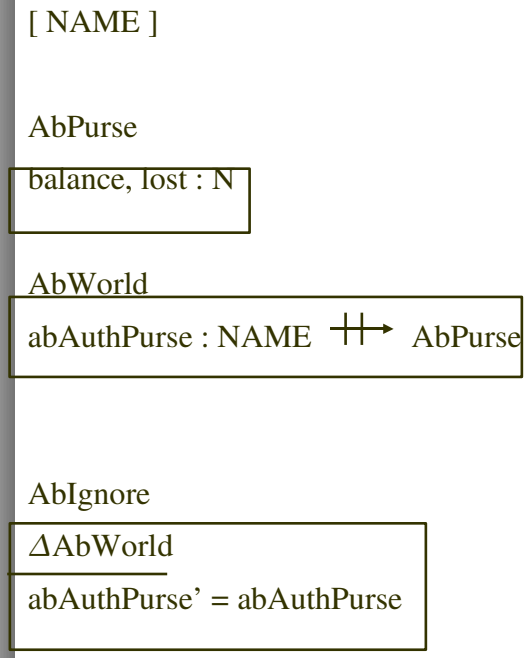
The identity of objects

- Z : schemas define records
- Alloy : signatures define atomic objects
  - Objects have an *identity*
    - Notion does not exist in Z
  - Suitable for names, coins
- Two objects with same field values may be distinct
  - Naive solution : impose equality constraint

```
fact {
    no disj a1,a2:AbPurse {
        a1.balance=a2.balance
        a1.lost=a2.lost
    }
}
```

The identity of objects

- Smoother solution : represent purses and states as standalone objects rather than records
  - No names




```
sig Coin

sig AbPurse {balance,lost: Coin->AbWorld}

sig AbWorld {abAuthPurse : set AbPurse}

pred AbIgnore (a,a':AbWorld) {
 a'.abAuthPurse = a.abAuthPurse
 all p : AbPurse | p in a.abAuthPurse implies {
   p.balance.a' = p.balance.a
   p.lost.a' = p.lost.a
 }
}
```

Outline

- Work progress since May
- Improving the Model
- Using FOL Theorem Provers
- Conclusion and Future Work

The direct attempt

- FOL atoms are Alloy atoms
  - But Alloy predicates take arbitrary relations as arguments
  - So they have to be inlined
  - Formulae become huge
- Simplifications to decrease formula size
  - Eliminate redundancy with subsumption tests
  - Split theorems through
  - Attempt to reach a normal form
    - Does not terminate
- Very few results :
  - Proved theorems relative to the abstract world (atomic transactions) alone

The “lifted” attempt

- FOL atoms are Alloy relations
- Axiomatize relational algebra
  - Bound arities according to spec in Alloy
- Problems :
  - Trouble to prove obvious-looking general theorems such as :
    - The Cartesian product of two atoms is a singleton of arity 2
  - Would have to prove intermediate lemmas
  - Loss of automation
- No significant results

Outline

- Work progress since May
- Improving the Model
- Using FOL Theorem Provers
- Conclusion and Future Work

Conclusion

- No further bugs found
- Scope issue not solved yet with Alloy Analyzer
  - Current scope increase with Kodkod ?
- But first proof attempts with FOL
  - Infiniteness still dropped
  - Very few results

Future work

- Argue small model theorem (Momtahan 2004) ?
- Improve checking with FOL theorem provers
  - To expect better FOL theorem provers is quite hopeless : undecidable
  - Better model Alloy into FOL
  - Fit into decidable sublogic ?
- Tackle finiteness
  - HOL necessary at first sight
  - Use incomplete FOL theories ?
- Interface Alloy method with others
  - May be feasible soon (cf. future Alloy workshop)

Acknowledgments

- At MIT :
  - The SDG group, in particular Daniel Jackson
  - But also the CRS group, in particular Viktor Kuncak and Charles Bouillaguet
- At ENS :
  - Patrick Cousot, who gave me the opportunity to follow the internship
- At RAL :
  - Jim Woodcock and Juan Bicarregui, for their hospitality

Any questions ?

- E-mail addresses
  - ramanana@mit.edu Tahina Ramananandro
  - dnj@mit.edu Daniel Jackson
- Alloy modules available at :
  - http://www.eleves.ens.fr/~ramanana/work/mondex
- Alloy Website :
  - http://alloy.mit.edu