A Feasibility Study on Using Classifying Terms in Alloy

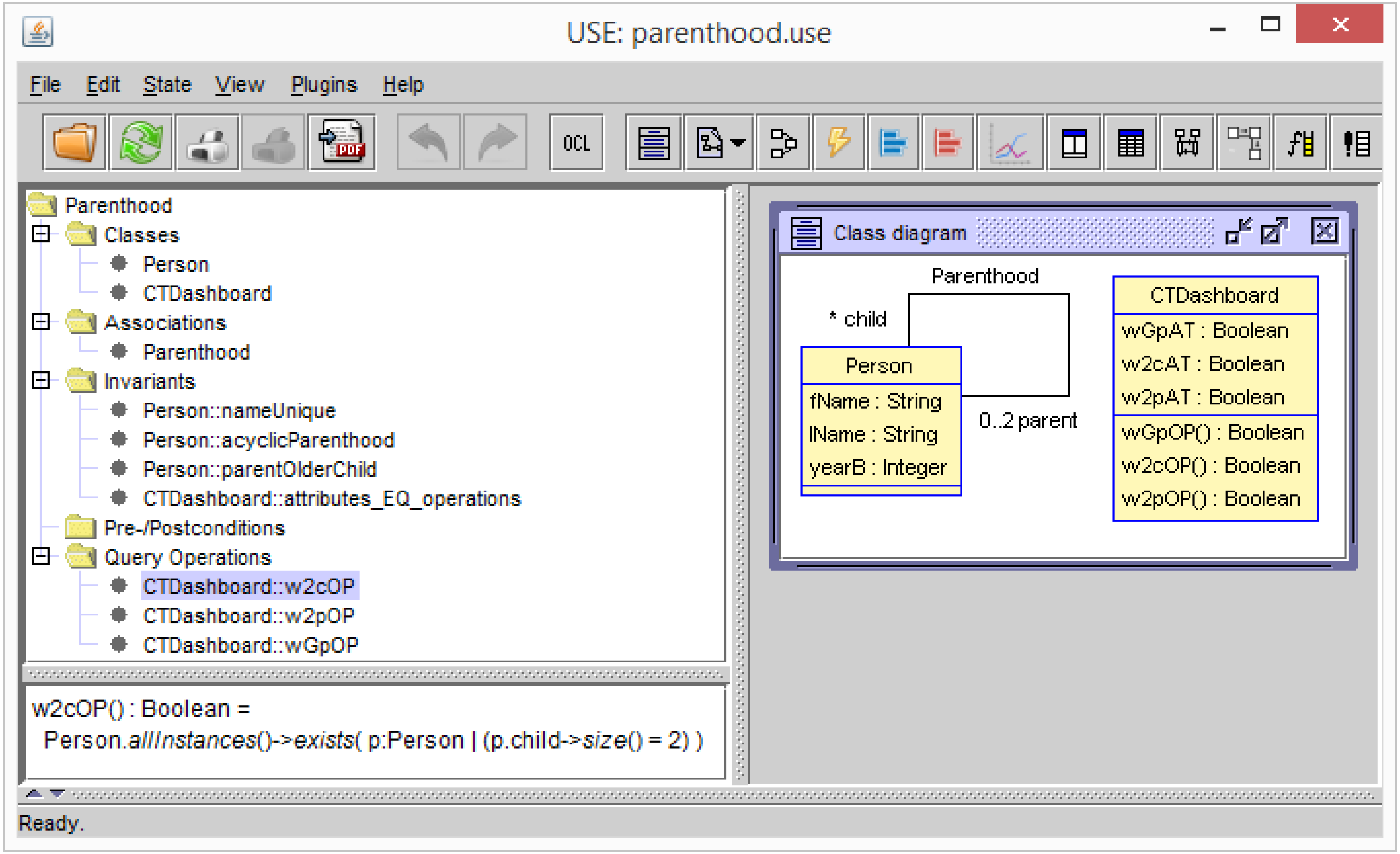
Robert Clarisó & Martin Gogolla

Universitat Oberta de Catalunya, Barcelona, Spain
University of Bremen, Bremen, Germany

Motivation and Overview on Approach

- Fundamental problem for class model / conceptual schema is satisfiability: finding an instantiation (object diagram) populating the model and fulfilling integrity constraints
- Model finders: finding such instantiations; output of model finder used for validation (instantiations as [counter-] examples); for testing purposes (instantiations as test cases); outputs should be diverse: representing wide range of scenarios and situations
- Premise here: classifying terms (CTs); designer gives collection of expressions to detect differences between two instantiations; CTs guide model finding to catch solutions being diverse by construction; developed for UML class diagrams annotated with OCL; implemented in USE
- Here: feasibility of using classifying terms for Alloy; also discuss limitations of Alloy from point of view of CTs; querying solutions: query expressions can only be evaluated interactively through GUI or programmatically by calling the Alloy API

Running Example



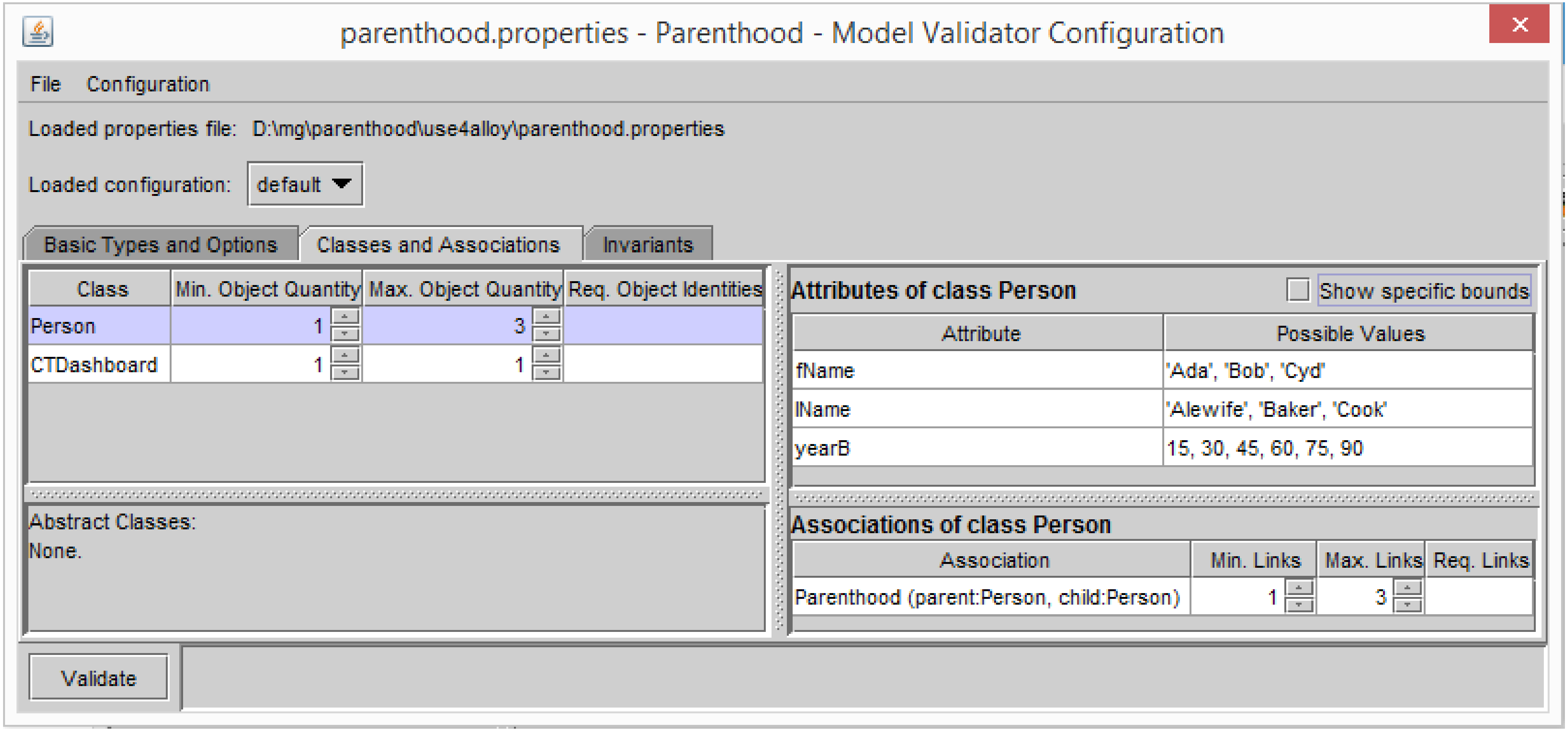
- Parenthood
  - Classes
    - Person
    - CTDashboard
  - Associations
    - Parenthood
  - Invariants
    - Person::nameUnique
    - Person::acyclicParenthood
    - Person::parentOlderChild
    - CTDashboard::attributes_EQ_operations
  - Pre-/Postconditions
  - Query Operations
    - CTDashboard::w2cOP
    - CTDashboard::w2pOP
    - CTDashboard::wGpOP

```
w2cOP() : Boolean =
  Person.allInstances()->exists( p:Person | (p.child->size() = 2) )
```

Invariants and Classifying Terms for Running Example

```
context p1,p2:Person inv nameUnique:
  p1<>p2 implies
   (p1.fName<>p2.fName or p1.lName<>p2.lName)

context p:Person inv acyclicParenthood:
  p.parent->closure(p | p.parent)->excludes(p)

context p:Person inv parentOlderChild:
  p.child->forAll(c | p.yearB+15 <= c.yearB)


wGp
  Person.allInstances->exists(g,p,c |
    g.child->includes(p) and p.child->includes(c))
w2c
  Person.allInstances->exists(p | p.child->size=2)
w2p
  Person.allInstances->exists(p | p.parent->size=2)
```

USE Model Validator Configuration



parenthood.properties - Parenthood - Model Validator Configuration

Loaded properties file: D:\mg\parenthood\use4alloy\parenthood.properties

Loaded configuration: default

Class	Min. Object Quantity	Max. Object Quantity	Req. Object Identities
Person	1	3	
CTDashboard	1	1	

Abstract Classes:
None.

Attributes of class Person

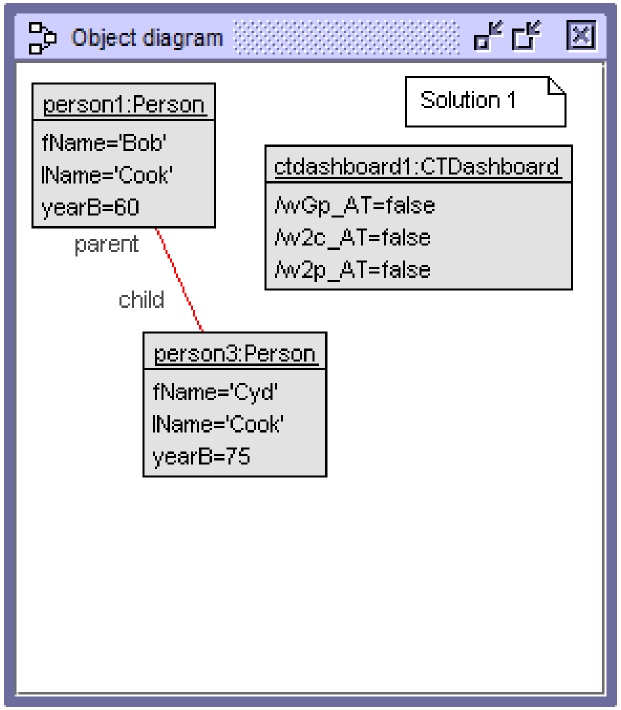
□ Show specific bounds

Attribute	Possible Values
fName	'Ada', 'Bob', 'Cyd'
lName	'Alewife', 'Baker', 'Cook'
yearB	15, 30, 45, 60, 75, 90

Associations of class Person

Association	Min. Links	Max. Links	Req. Links
Parenthood (parent:Person, child:Person)	1	3	

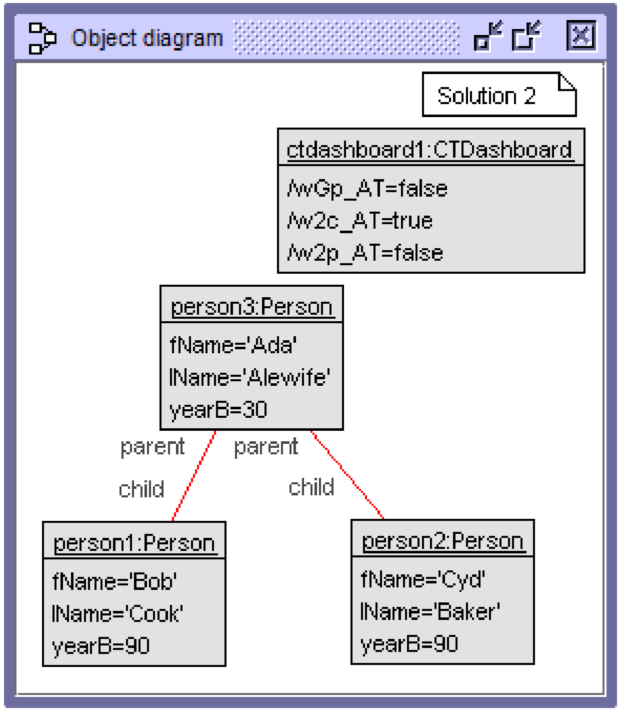
Solutions Found by USE Model Validator



person1:Person
fName='Bob'
lName='Cook'
yearB=60

ctdashboard1:CTDashboard
/wGp_AT=false
/w2c_AT=false
/w2p_AT=false

person3:Person
fName='Cyd'
lName='Cook'
yearB=75

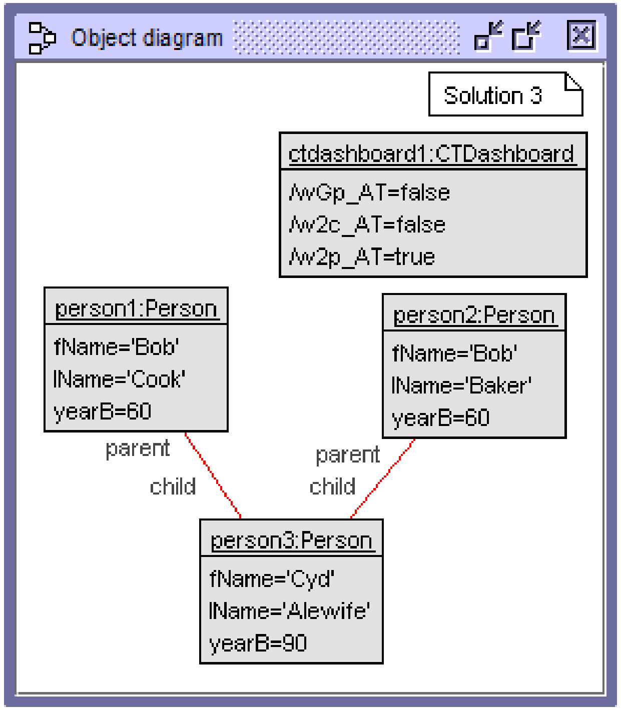


ctdashboard1:CTDashboard
/wGp_AT=false
/w2c_AT=true
/w2p_AT=false

person3:Person
fName='Ada'
lName='Alewife'
yearB=30

person1:Person
fName='Bob'
lName='Cook'
yearB=90

person2:Person
fName='Cyd'
lName='Baker'
yearB=90

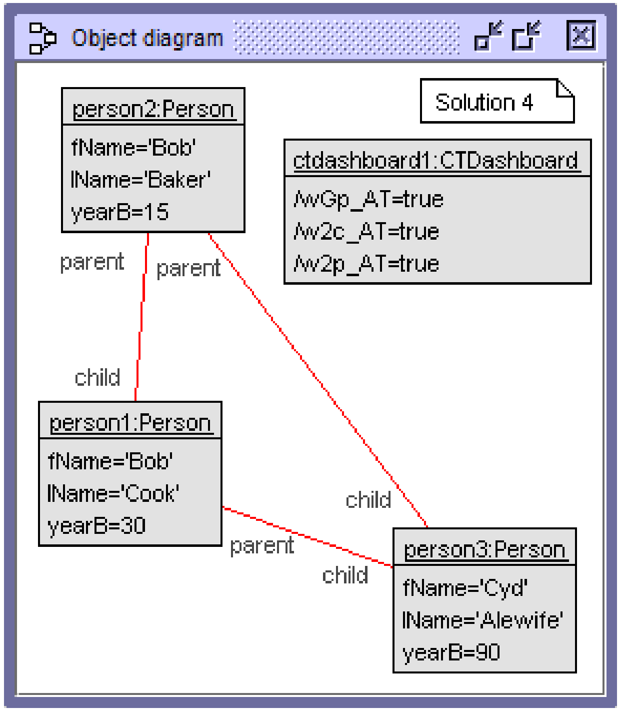


ctdashboard1:CTDashboard
/wGp_AT=false
/w2c_AT=false
/w2p_AT=true

person1:Person
fName='Bob'
lName='Cook'
yearB=60

person2:Person
fName='Bob'
lName='Baker'
yearB=60

person3:Person
fName='Cyd'
lName='Alewife'
yearB=90

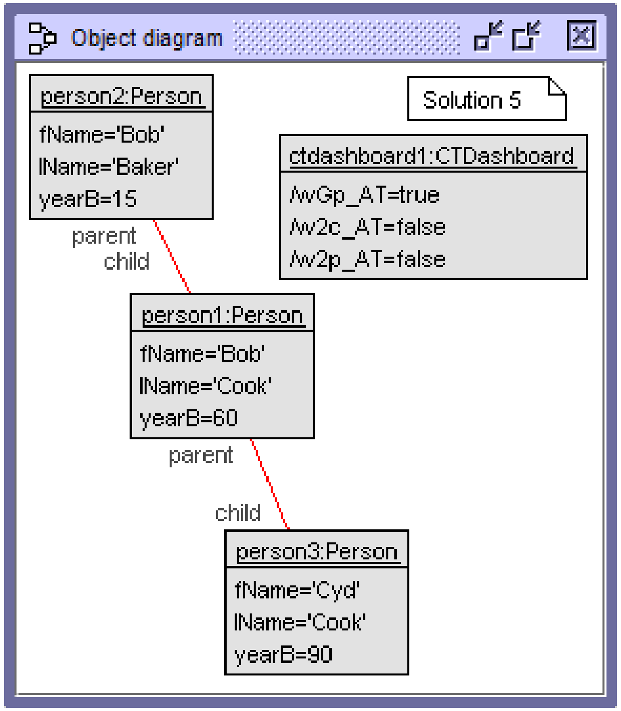


person2:Person
fName='Bob'
lName='Baker'
yearB=15

ctdashboard1:CTDashboard
/wGp_AT=true
/w2c_AT=true
/w2p_AT=true

person1:Person
fName='Bob'
lName='Cook'
yearB=30

person3:Person
fName='Cyd'
lName='Alewife'
yearB=90



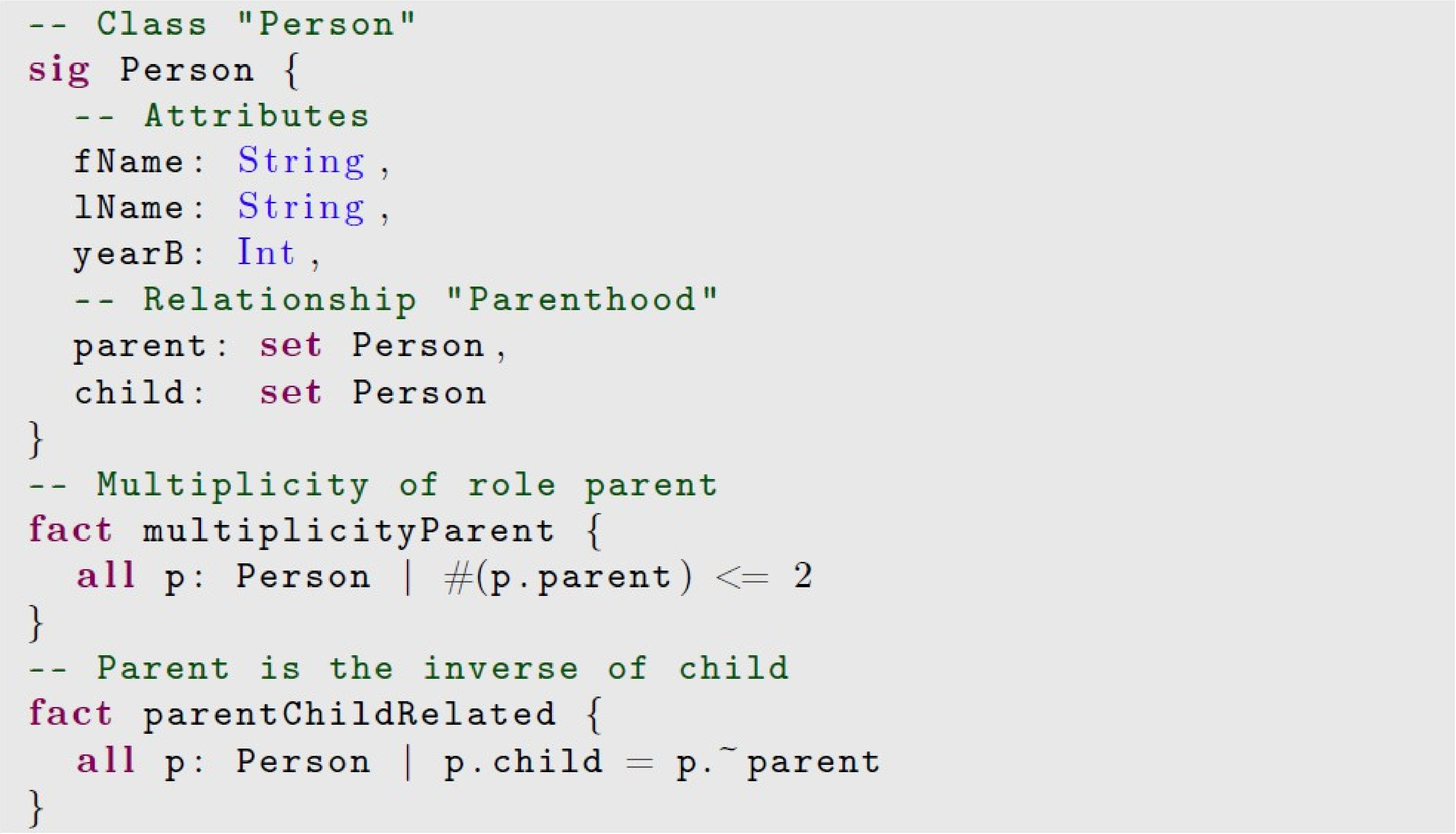
person2:Person
fName='Bob'
lName='Baker'
yearB=15

ctdashboard1:CTDashboard
/wGp_AT=true
/w2c_AT=false
/w2p_AT=false

person1:Person
fName='Bob'
lName='Cook'
yearB=60

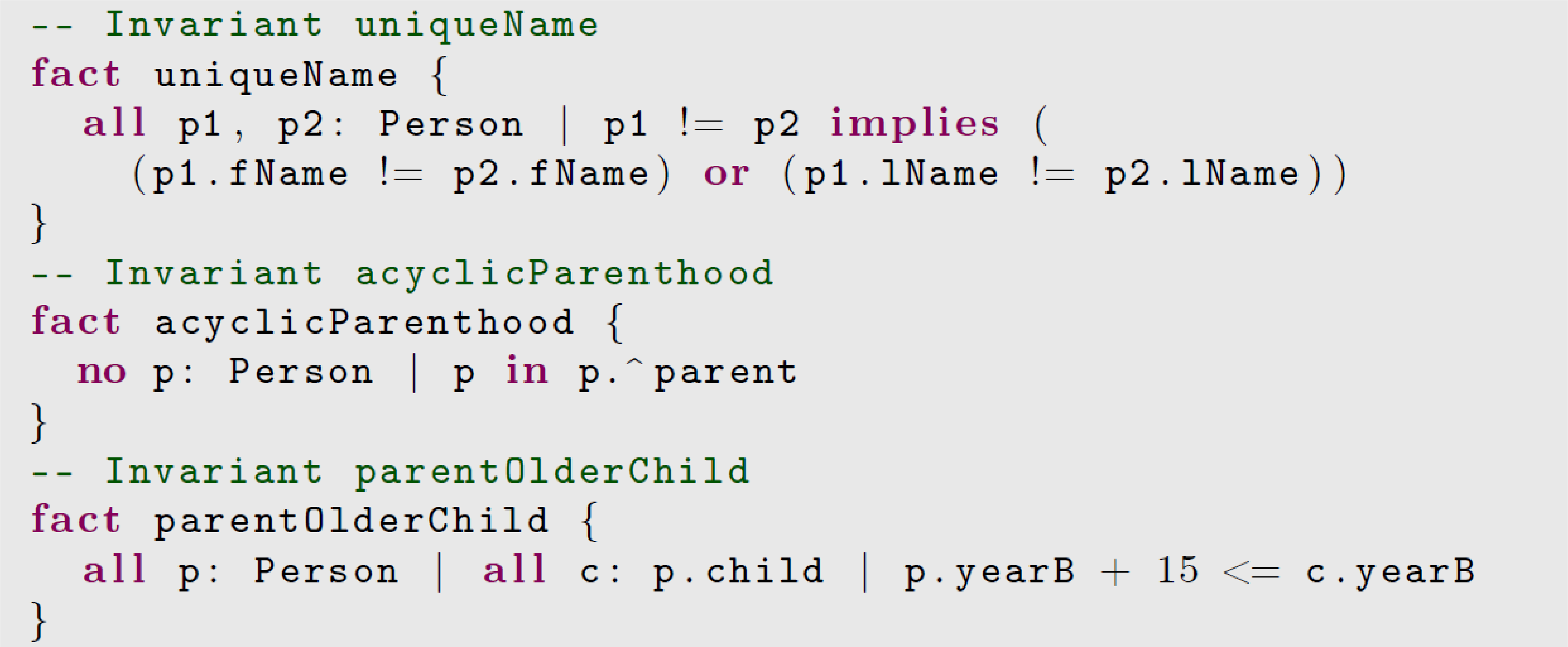
person3:Person
fName='Cyd'
lName='Cook'
yearB=90

Formulation of Example Class Diagram in Alloy



```
-- Class "Person"
sig Person {
  -- Attributes
  fName: String,
  lName: String,
  yearB: Int,
  -- Relationship "Parenthood"
  parent: set Person,
  child:  set Person
}
-- Multiplicity of role parent
fact multiplicityParent {
  all p: Person | #(p.parent) <= 2
}
-- Parent is the inverse of child
fact parentChildRelated {
  all p: Person | p.child = p.~parent
}
```

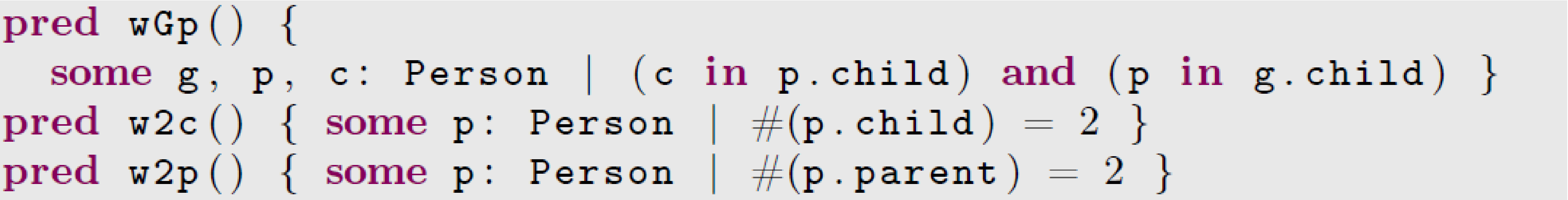
Formulation of Invariants in Alloy



```
-- Invariant uniqueName
fact uniqueName {
  all p1, p2: Person | p1 != p2 implies (
    (p1.fName != p2.fName) or (p1.lName != p2.lName))
}
-- Invariant acyclicParenthood
fact acyclicParenthood {
  no p: Person | p in p.^parent
}
-- Invariant parentOlderChild
fact parentOlderChild {
  all p: Person | all c: p.child | p.yearB + 15 <= c.yearB
}
```

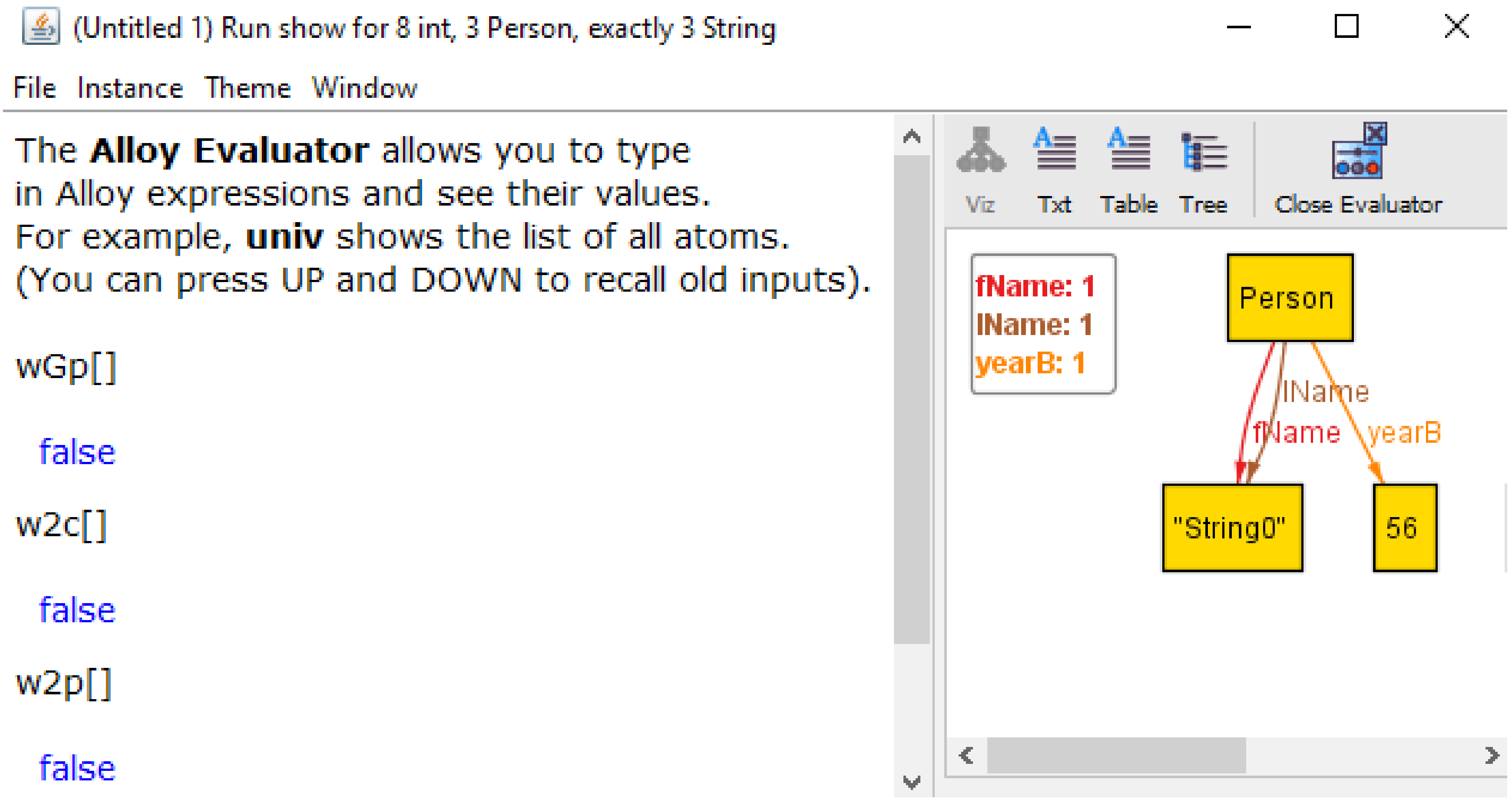
Formulation of CTs and Simulating their Evaluation in Alloy

1. Defining classifying terms.
2. Finding a valid instantiation.
3. Evaluating classifying terms on a given instantiation.
4. Defining a new invariant for our model.



```
pred wGp() {
  some g, p, c: Person | (c in p.child) and (p in g.child) }
pred w2c() { some p: Person | #(p.child) = 2 }
pred w2p() { some p: Person | #(p.parent) = 2 }
```

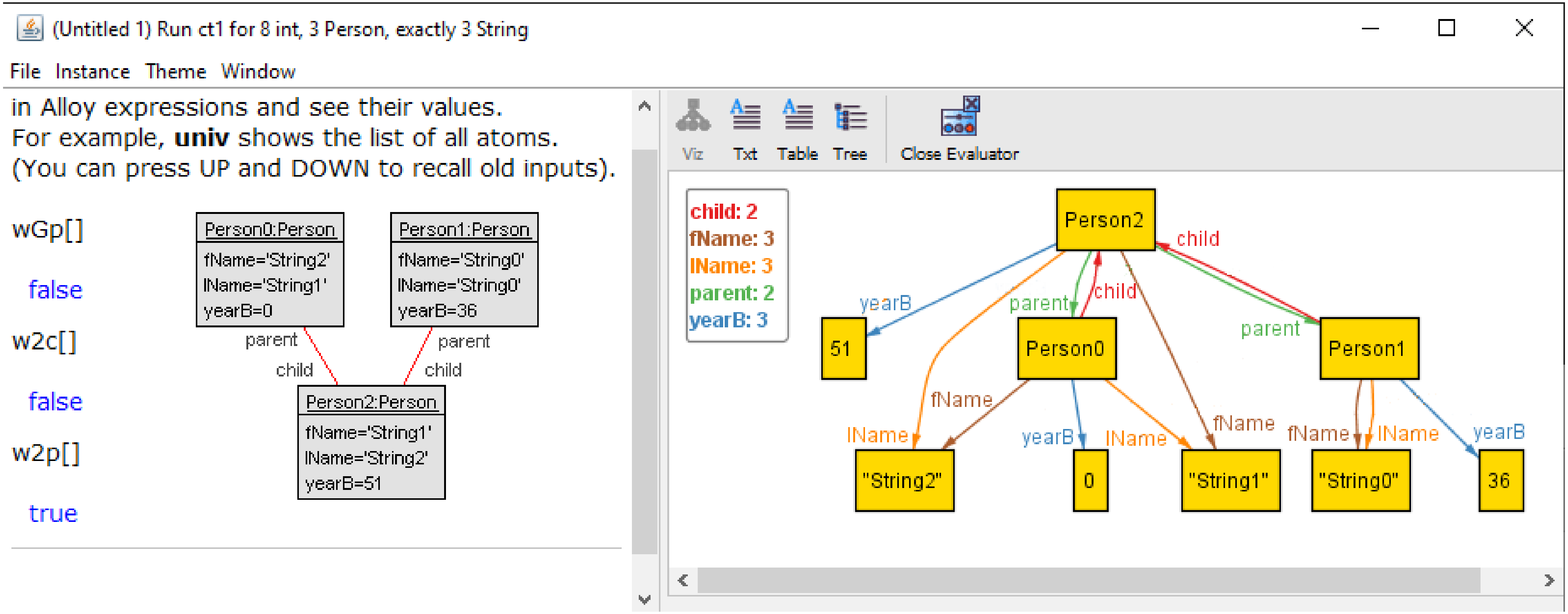
First Solution by Alloy (same equivalence class as USE solution 1)



```
The Alloy Evaluator allows you to type
in Alloy expressions and see their values.
For example, univ shows the list of all atoms.
(You can press UP and DOWN to recall old inputs).

wGp[]

  false

w2c[]

  false

w2p[]

  false
```

```
pred forbidFFF() {
  not (not wGp[] and not w2c[] and not w2p[]) }
```

Second Solution by Alloy (same equivalence class as USE solution 3)



(Untitled 1) Run ct1 for 8 int, 3 Person, exactly 3 String

in Alloy expressions and see their values.
For example, **univ** shows the list of all atoms.
(You can press UP and DOWN to recall old inputs).

```
wGp[]
  false
w2c[]
  false
w2p[]
  true
```

Person0:Person
fName='String2'
lName='String1'
yearB=0

Person1:Person
fName='String0'
lName='String0'
yearB=36

Person2:Person
fName='String1'
lName='String2'
yearB=51

Conclusions and Future Work

- Proposed strategy for applying classifying terms in Alloy; CTs used to control output of Alloy Analyzer and to ensure diversity of generated instantiations
- First output instantiation, then change and add commands after each output; in output instantiation, assess values of classifying term, define new predicate adding new constraint: combination of values for classifying terms obtained by last command now forbidden; ensuring next instantiation differs in the value of at least one classifying term from preceding outputs; continue until no further output instantiation is found
- Future work: automate approach; implement it in Alloy, so overall process performed automatically
- Consider other textual modeling approaches like B, Event-B, SQL: checking whether idea of classifying terms can be applied

Thanks for your attention!