Alloy Operators

Curt Clifton
Rose-Hulman Institute of Technology

Constants

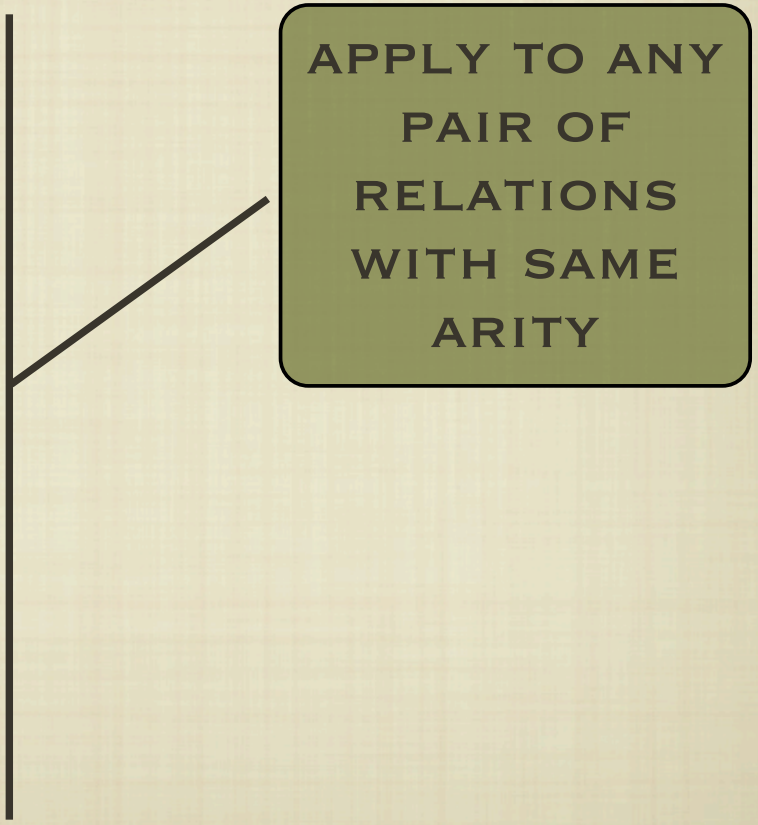
- **none** – the empty set
- **univ** – the universal set
  - Contains one of every atom in the instance
- **iden** – the identity relation
  - Maps every atom to itself
  - Including Int atoms -8, -7, ... 6, 7

In every instance

Q1,2

Set Operators

- Set valued:
  - + – union
  - & – intersection
  - - – difference
- Boolean valued:
  - in – subset
  - = – equality



Q3

Relational Operators

- Combining relations
  - -> – arrow product
  - . – dot join
  - [] – box join
- Reachability
  - ^ – transitive closure
  - * – reflexive-transitive closure
- "Modifying" relations
  - ~ – transpose
  - <: – domain restriction
  - :> – range restriction
  - ++ – override

Combining Relations: Arrow Product

- `p -> q` – every possible way to concatenate a tuple from P and a tuple from Q

Q4

Combining Relations: Dot Join

- Also called "composition"
- Most important Alloy operator
- Written: p.q
- Examples...

Dot Join on Tuples

- {(NO,**AO**)} . {(**AO**,DO)} = {(NO,DO)}
- {(NO,**DO**)} . {(**NO**,DO)} = {}
- {(NO,**DO**)} . {(**D1**)} = {}
- {(NO,**DO**)} . {(**DO**)} = {(NO)}
- {(**NO**)} . {(**NO**,DO)} = {(DO)}
- {(**BO**)}.{(**BO**,NO,DO)} = {(NO,DO)}

Q5

Dot Joins on Relations

- Suppose **p** and **q** are relations
- **p.q** is the result of taking every combination of a tuple from **p** and a tuple from **q** and including their join

Cartoon of the Day

Combining Relations: Box Join

- Exactly the same meaning as dot join
- Syntactic sugar—looks like array access
- Written: q[p], means p.q
  - Typically p will be a scalar

"Modifying" Relations: Transpose

- ~p – transpose
  - Yields a new relation by flipping order of tuples in p
  - p must be binary
  - Useful for getting order right before a join

Example: of whom is Name$2 a friend

Q7

"Modifying" Relations: Restrict domain/range

- s <: r – domain restriction
  - Yields a new relation by throwing out tuples in r that don't start with an element of s
  - s must be a set
  - Useful for focusing on "subclasses"
- r :> s – range restriction, like domain restriction but matches last element of r's tuples

"Modifying" Relations: Override

- `p ++ q` – override
  - Like union, but tuples in q replace "matching" tuples from p
  - "Matching" means first elements are the same
  - Yields a new relation
  - `p` and `q` must have same arity
  - Useful for modeling mutation

Q8

Next Time

- Reachability operators
- Constraints

Model from class

```
sig Book {
        contacts: Name -> lone Addr,
        friends: set Name,
        owner: lone Name
}
sig Name, Addr {}

pred show[b1, b2:Book] {
        b1 != b2
        #b1.contacts > 1
        #b2.contacts > 1
}

run show for 3 but 2 Book

pred addContact[b, b': Book, n: Name, a: Addr] {
        b'.contacts = b.contacts ++ n->a
}

pred addContacts[b, b': Book, c: Name -> lone Addr] {
        b != b'
        b'.contacts = b.contacts ++ c
}

run addContacts for 4 but 2 Book
```