CIS 771:
Software Specifications

Lecture: Alloy Whirlwind Tour (part A)

Alloy Quick Tour

Objectives of the Quick Tour lectures

- Understand primary features of the Alloy modeling language
  - modeling structures
  - specifying constraints
- Understand the basic capabilities of the Alloy Constraint Analyzer (ACA) automated tool
- Be able to write Alloy specifications that model simple systems
- Be able to run ACA to analyze simple systems

…this short lecture gives a quick tour of the quick tour

Address Book Application

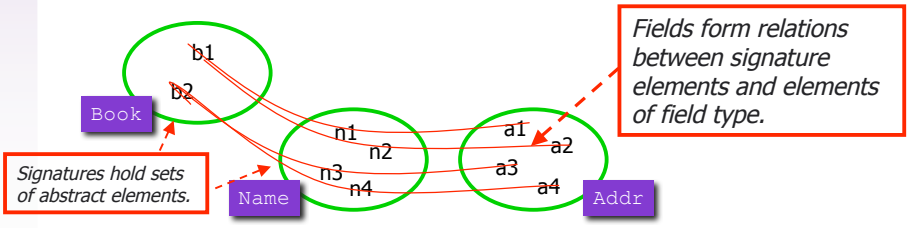
Presentation will be based on a simple application

- Name
- Address
- Lists

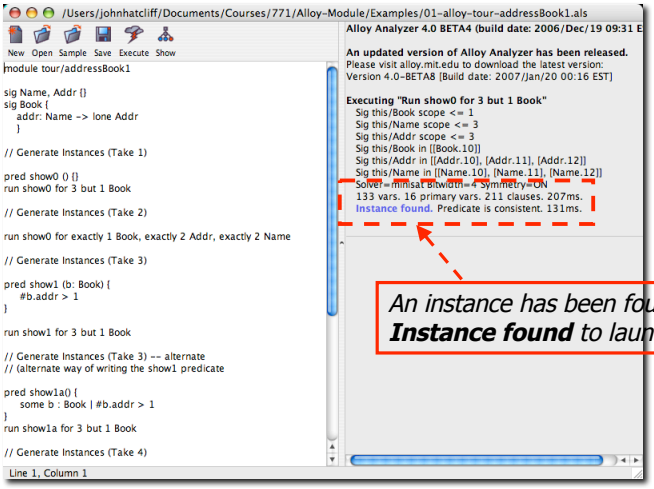
Text-based Modeling Language

```
module tour/addressBook1

    sig Name, Addr {}
    sig Book {
          addr: Name -> lone Addr
    }
```

Formal semantics based on sets and relations…

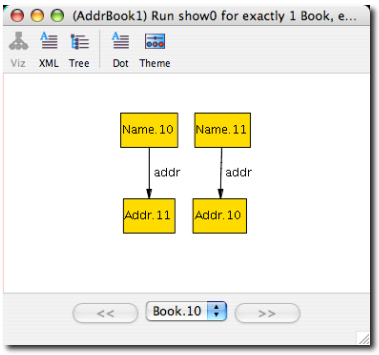


User Interface



Results

Result of execution from previous slide…



Settings: Goto Theme, then Book, then turn on *Project over this sig*

- Alloy automatically looks for system configurations that satisfy the specifications
- If we expected instances, but none can be found we know we have made a mistake by over-constraining our specification
- If instances are found that we didn't expect, this may mean that we need to add more constraints to our specification to rule out undesirable configurations

Generating Instances

Add constraints to direct the analysis…

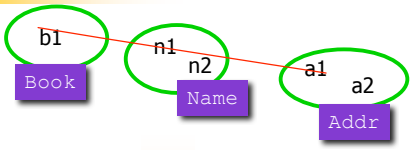
```
pred show1 (b: Book) {
 #b.addr > 1
 }

run show1 for 3 but 1 Book
```

Introduce a constraint that requires the existence of some Book b, such that the number of entries in the addr map for b is > 1.

Example Operations

Pre-State

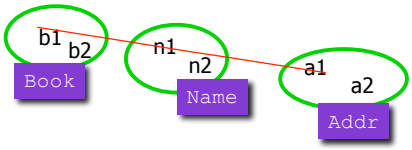
…adding a new book



*Operational view: Operation **transforms** the pre-state into the post-state*

*Declarative view: Operation **relates** the pre-state and the post-state*

Post-State



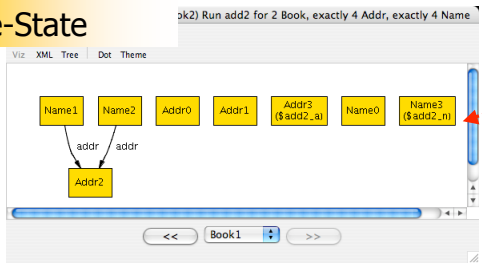
Modeling *Add Address*

```
pred add2(b, b': Book, n:Name, a:Addr) {
  // pre-condition
  // n is not currently in the book
  no n.(b.addr)

  // post-condition
  // invoking b' addr map on n yield's a
  n.(b'.addr) = a

  // frame-condition
  // for all other names, the addr map should
  // yield the same value
  all n1: (Name - n) | n1.(b.addr) = n1.(b'.addr)
}
```


Results

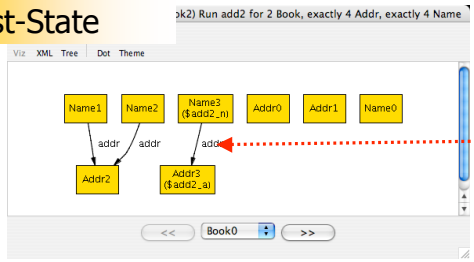
using project over Book sig option in visualization

Pre-State



Name not associated with Addr

Post-State

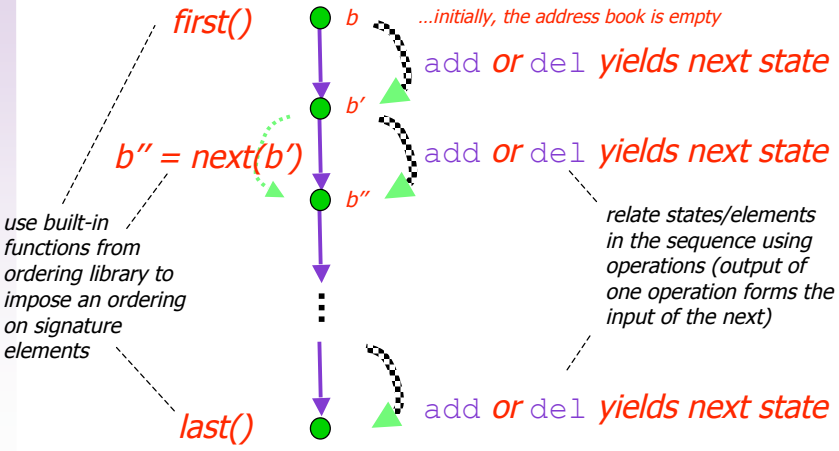


Name maps to Addr

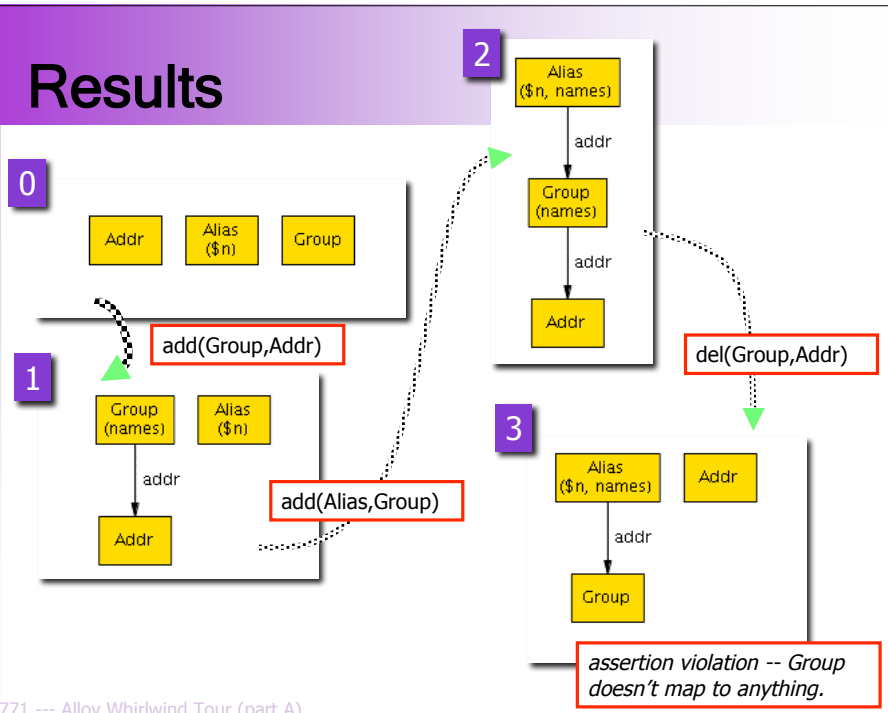
```
run add2 for 2 Book, exactly 4 Name, exactly 4 Addr
```

Modeling System Executions

Consider the address book example…



Results



Acknowledgements

- The material in this lecture is based on Chapter 2 from…
  - *Software Abstractions: Logic, Language, and Analysis,* Daniel Jackson, MIT Press, 2006.