Verifying Behavioral Subtyping in TVLA

Anne Mulhern
UW Madison
mulhern@cs.wisc.edu

Overview

- Subtyping and Subclassing
- Behavioral Subtyping and Structural Subtyping
- Verifying Behavioral Subtyping in TVLA
  - Method Equivalence
  - TVLA Techniques
- Work So Far
- Future Work

Subtyping vs. Subclassing

- Inheritance of code, i.e. subclassing
- Inheritance of behavior, i.e. subtyping
- Liskov Substitution Principle:

  For every object x' of type t' there is an object x of type t, such that for all programs P defined in terms of t, the behavior of P is unchanged when x' is substituted for x. [Liskov 1988]

- Subtyping not enforced by compilers
- Goal: Build a tool that provides some amount of checking

Why?

```
class FooNode {
  FooNode next;
  . . .
  ⟨many data members⟩
  . . .
};
```

≅

```
class ListNode {
  ListNode next;
};
```

```
class Foo {
  FooNode first;
  FooNode last;
  AppendElmt(Datum);
  . . .
  ⟨many members⟩
  . . .
};
```

```
class List {
  ListNode first;
  ListNode last;
  AddToEnd(Datum);
};
```

Related Work

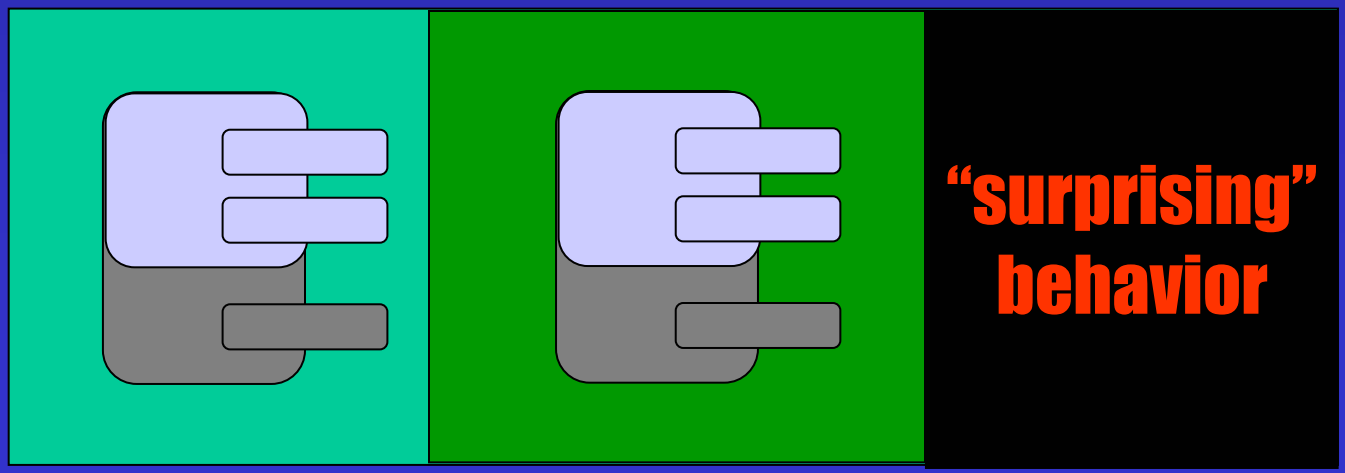
- Liskov & Wing
  - A Behavioral Notion of Subtyping [1994]
  - Behavioral Subtyping Using Invariants and Constraints [1999]
- America:
  - Designing an Object-Oriented Programming Language with Behavioral Subtyping [1991]
- Leavens & Dhara:
  - Weak Behavioral Subtyping for Types With Mutable Objects [1994]
- Findler, Latendresse & Felleisen:
  - Behavioral Contracts and Behavioral Subtyping [2001]
- Findler & Felleisen:
  - Contract Soundness for Object-Oriented Languages [2001]

Overview

- Subtyping vs. Subclassing
- Behavioral Subtyping and Structural Subtyping
- Verifying Behavioral Subtyping in TVLA
  - Method Equivalence
  - TVLA Techniques
- Work So Far
- Future Work

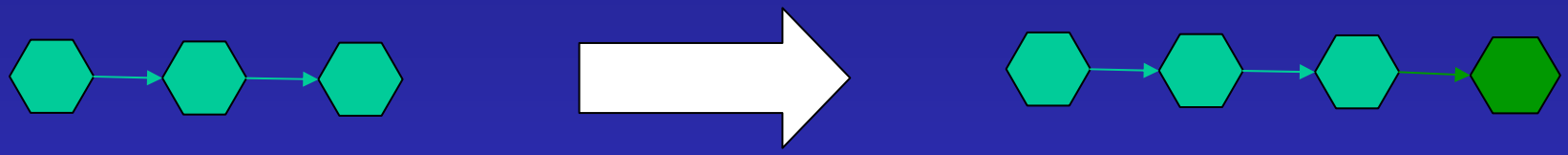
Behavioral Subtyping

- Sometimes difficult to define what is subtyping
- “behavior” hard to specify
- Typically relies on some programmer specifications
- A subtype object is sometimes operated on by supertype methods and sometimes by subtype methods (polymorphism problem)

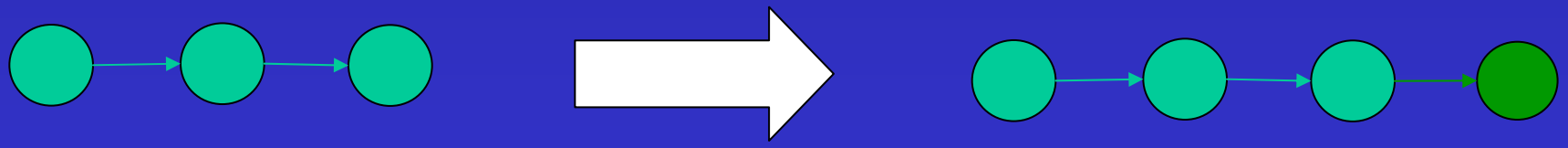


Structural Subtyping

- Are the structures substitutable?



f.AppendElmt(Datum);



l.AddToEnd(Datum);

Goals

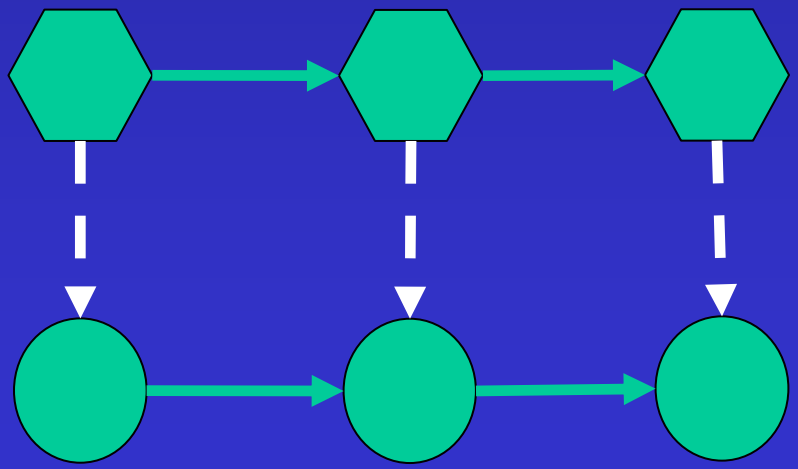
- Keep programmer input to a minimum
  - Example: programmer asserts correspondence
    - Between fields of the class
    - Between methods of the class
    - Rest is up to tool
- Need to verify
  - Method equivalence: corresponding methods of the subclass and superclass do the "same" thing
  - If "new" methods of the subclass are executed surprising behavior won't occur subsequently

Overview

- Subtyping vs. Subclassing
- Behavioral Subtyping and Structural Subtyping
- Verifying Behavioral Subtyping in TVLA
  - Method Equivalence
  - TVLA Techniques
- Work So Far
- Future Work

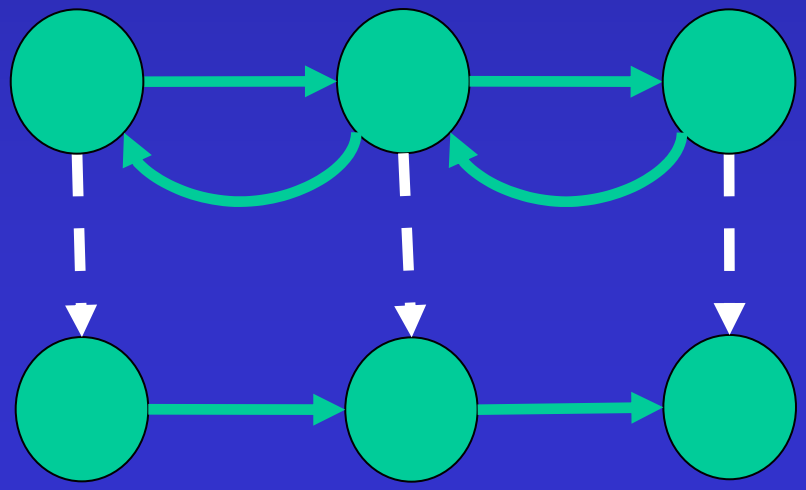
How?

- TVLA (Three valued logic analysis)
- Models the different elements in the data structure
- Models equivalence between two structures
  - Maintains a correspondence between elements



Example

- Assert: doubly linked list is a subclass of a singly linked list
- The back pointer in a doubly linked list stands in for the additional fields defined in Foo

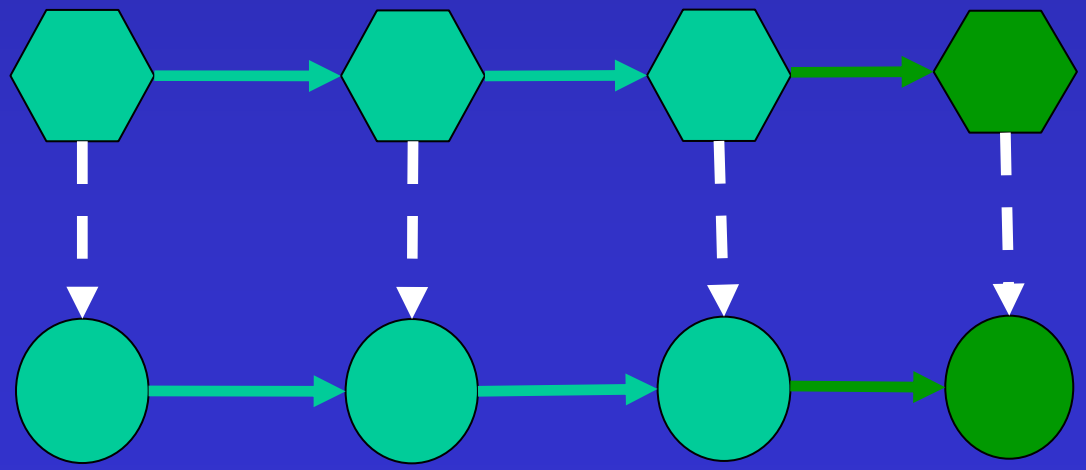


Overview

- Subtyping vs. Subclassing
- Behavioral Subtyping and Structural Subtyping
- Verifying Behavioral Subtyping in TVLA
  - Method Equivalence
  - TVLA Techniques
- Work So Far
- Future Work

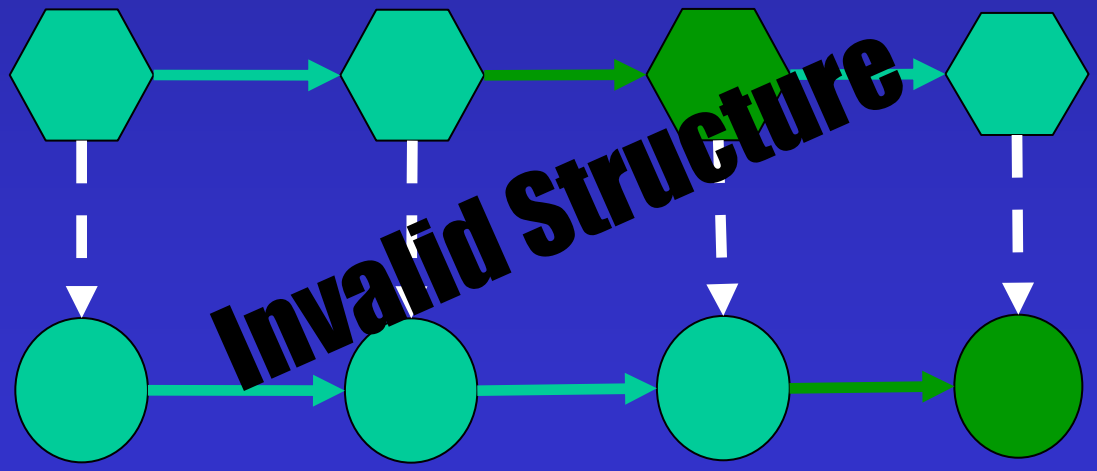
Method Equivalence

- General idea:
  - Invoke "equivalent" methods simultaneously on corresponding structures
    - Maintain correspondence between nodes
    - If correspondence is maintained throughout then structures are equivalent.



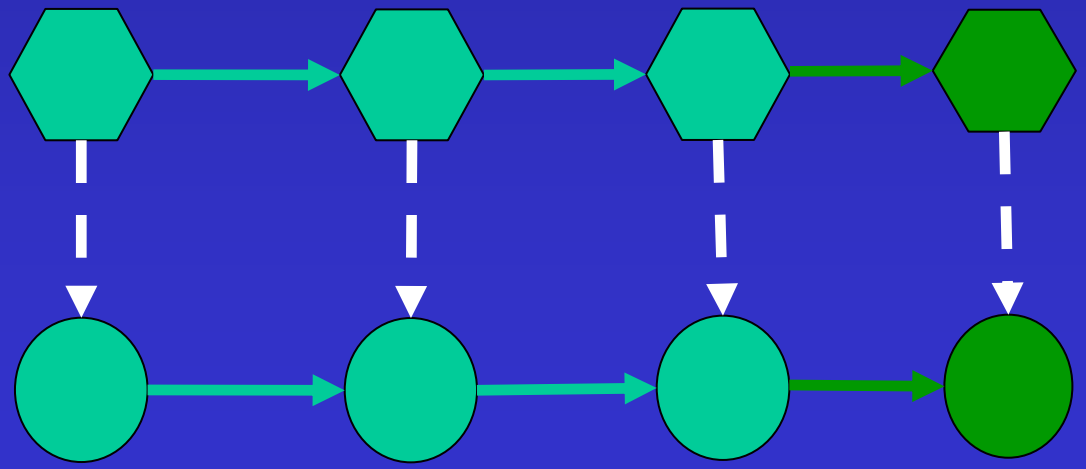
Method Equivalence

- Problem I:
  - TVLA structures must be detailed enough to capture the meaning of the operation
    - Example: AddToEnd must add element at end



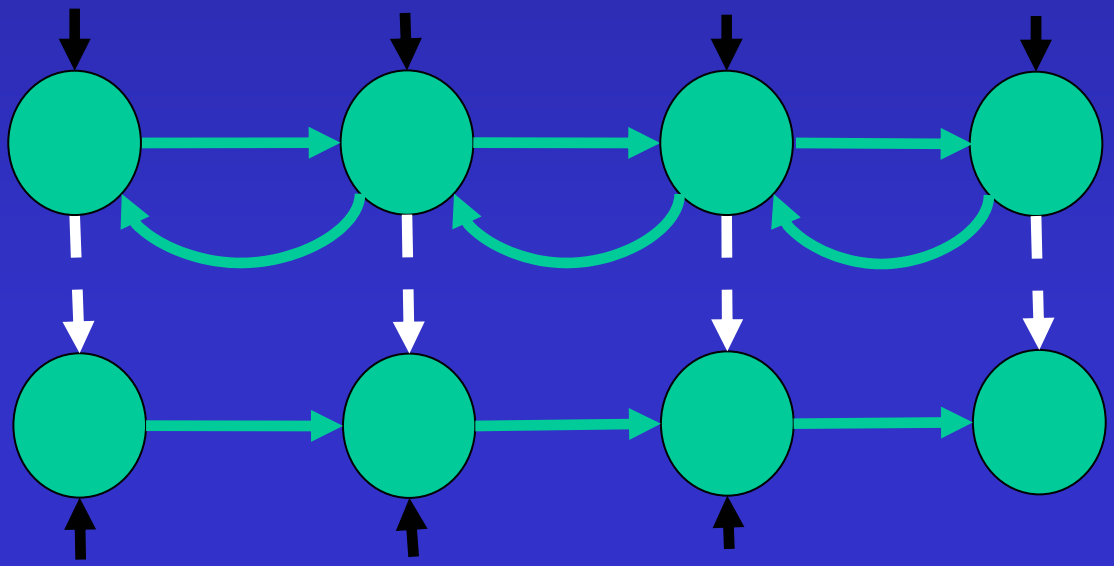
Method Equivalence

- Problem I Solution:
  - “Brand” specific nodes that will be affected.
    - Works only when a specific node can be designated.
    - May require programmer input.



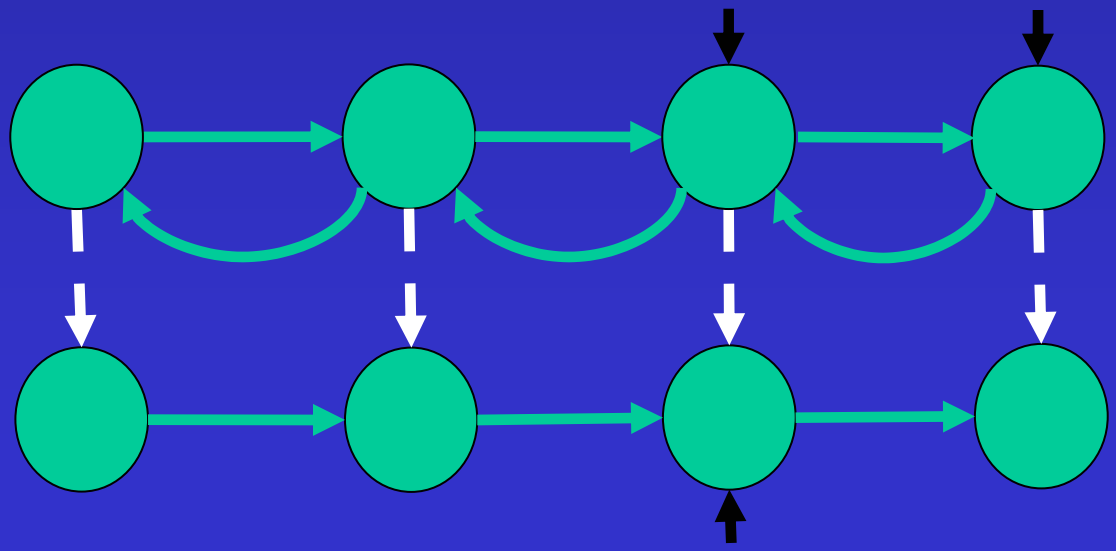
Method Equivalence

- Problem II:
  - Generally, equivalent methods will have very different implementations.
    - Example: Remove the last element from a list.
      - Doubly linked list can use back pointers.



Method Equivalence

- Problem II Solution:
  - Relax the requirement that the two structures always be in sync.
    - The structures must still be in sync at the start and end of the method.

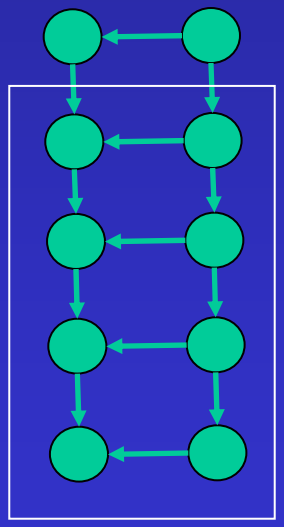


Overview

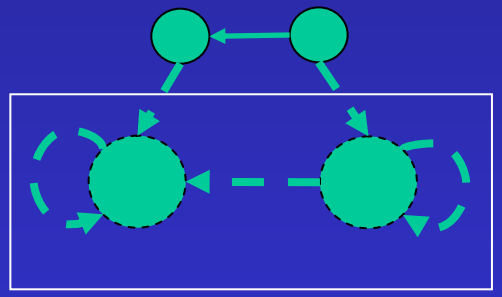
- Subtyping vs. Subclassing
- Behavioral Subtyping and Structural Subtyping
- Verifying Behavioral Subtyping in TVLA
  - Method Equivalence
  - TVLA Techniques
- Work So Far
- Future Work

Sharpening Predicates

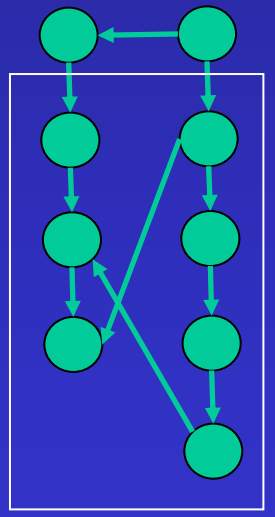
- Use a binary predicate to maintain the correspondence between a pair of nodes.
- Summarization results in loss of precision.



before summarization



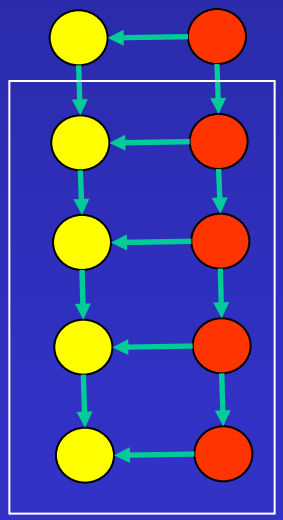
summarized



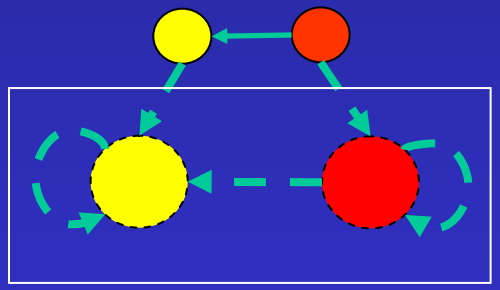
one possible structure

Sharpening Predicates

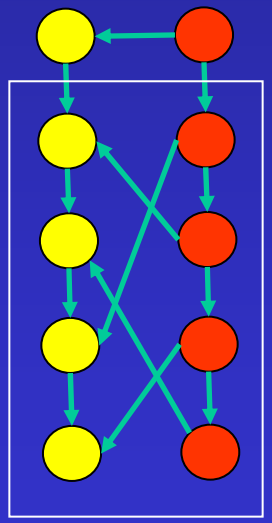
- Add a unary predicate that indicates the existence of the binary predicate.
- Each node definitely has a partner.



before summarization



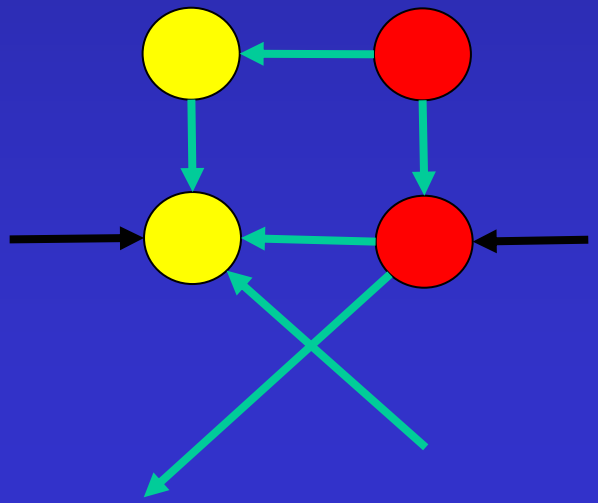
summarized



one possible structure

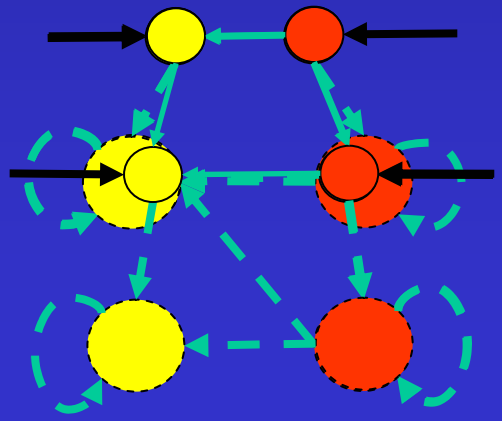
Sharpening Predicates

- Define the correspondence "recursively"
- If
  - the previous two nodes correspond, and
  - each of the current nodes has a partner,
- then the two current nodes correspond to each other.



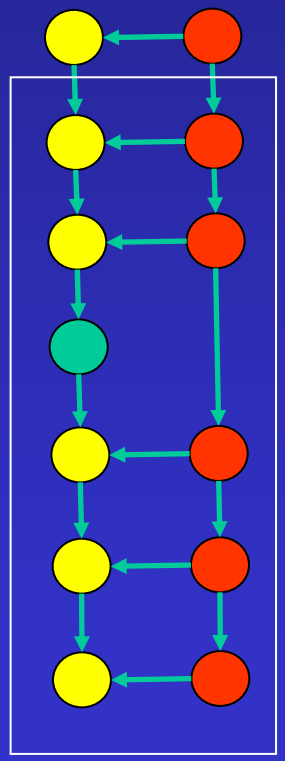
Sharpening Predicates

- Techniques
  - Strengthening the binary predicate with auxiliary unary predicates.
  - Making the definition recursive.
- Can resolve the correspondence while advancing in the list.

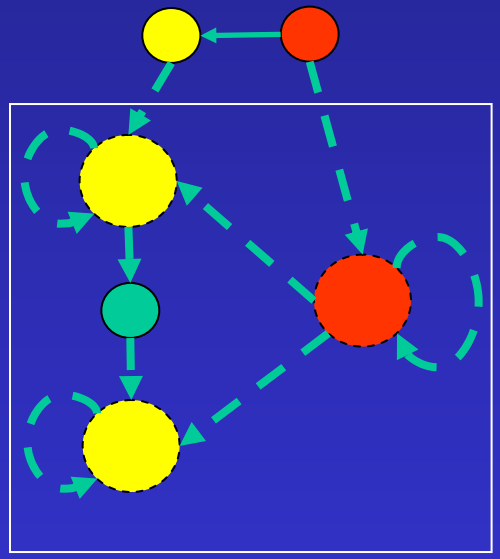


Sharpening Predicates

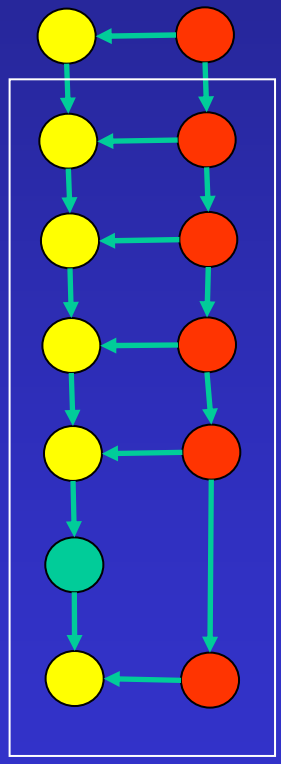
- Unary predicates prevent blurring of coupled and uncoupled nodes.



before summarization



summarized



one possible structure

Example

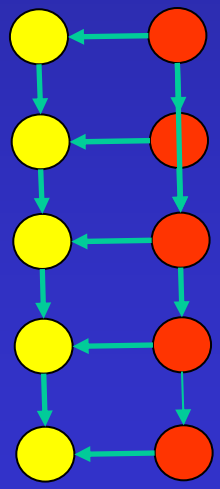
Remove Last Element

Overview

- Subtyping vs. Subclassing
- Behavioral Subtyping and Structural Subtyping
- Verifying Behavioral Subtyping in TVLA
  - Method Equivalence
  - TVLA Techniques
- Work So Far
- Future Work

Work so far

- Changing the structure of linked lists.
  - Inserting nodes.
  - Removing nodes.



Overview

- Subtyping vs. Subclassing
- Behavioral Subtyping and Structural Subtyping
- Verifying Behavioral Subtyping in TVLA
  - Method Equivalence
  - TVLA Techniques
- Work So Far
- Future Work

Future Work

- Generalize techniques to other problems.
  - Different structures, e.g. trees rather than lists
  - Behavior that is not just structural
    - e.g. do the methods order the nodes in the same way?
    - Build on previous work in TVLA on sorting.
- Enhance techniques
  - Numeric abstraction (attach integer values to nodes)
    - Can maintain a count of the number of non-summary nodes that a summary node represents
    - Can maintain more information about the position of a node within a structure