Mining Temporal Relationships between Data Invariants

Caroline Lemieux
Department of Computer Science
University of British Columbia

Program specifications (specs) are useful
Developers rarely write down program specs
• specifications can be tedious to specify manually
• may fall out of date quickly

Spec inference: likely specs without manual effort

Temporal Specs

Data Specs

• enqueue() is always followed by dequeue()

enqueue(5)
enqueue(1)
enqueue(24)
enqueue(7)
enqueue(19)
dequeue()
dequeue()

at exit of
deq(): size == 1
enq(): size == 2
enq(): size == 1
enq(): exit
size == 0

But: data values may persist or interact through time

data-temporal specifications

size >= 1
size == 0
size == 2
size == 1
size == 2
size == 1
size == 1
size == 0

provide more information than data specs or temporal specs alone

Temporal invariants

• program maintenance
• confirm expected behavior
• bug detection
• test generation
• system comprehension
• system modeling
• reverse engineering

Preliminary evaluation

Ran Quarry on QueueAr, a queue implemented with a wrap-around array.

Ongoing work 1

Does “size >= 3” always hold on this trace?

current string semantics: no

currentSize == 3
data invariant semantics: yes

currentSize == 3
size == 4
size == 3
size == 4
size == 3

Future work: incorporate SMT/theorem proving tools

Ongoing work 2

Quarry mined 100s of spec instances on QueueAr

Future work: design interestingness filter