Collecting invariants of data structures

Alessandro Schena

Constructor institute of technology

September 3, 2024

Article yet to be published

Content

Collect the class invariants of various data structures.

Set aside all problems regarding the consistency of aggregate objects.

Motivation

Have a formal description of data structures.

Improve quality of automatically generated code.

How did we group the data structures?

Access

- Dictionary
- Stack
- Queue
- Deque

Storage

- Linked-lists
- Hash-tables
- Trees

Example: Stack

```
deferred class STACK [G]
feature
  push (v: G)
    deferred
    ensure
      sequence.first = v
      sequence.but_first = (old sequence)
    end
  pop: G
    require
      sequence.count > 0
    deferred
    ensure
      Result = (old sequence).first
      sequence = (old sequence).but_first
    end
  sequence: SEQUENCE [G]
invariant
  sequence.count ≥ 0
end
```

Example: Linked lists

```
class LINKED_LIST_CELL [G] feature
  key: G
  next: detachable like Current
invariant
  next ≠ Current
end


class DOUBLY_LINKED_LIST_CELL [G] inherit LINKED_LIST_CELL [G] feature
  previous: detachable like Current
invariant
  previous ≠ Current
  (next ≠ Void) ⇒ next.previous = Current
  (previous ≠ Void) ⇒ previous.next = Current
end


class CIRCULAR_LINKED_LIST_CELL [G] inherit DOUBLY_LINKED_LIST_CELL [G] feature
  cycle: SEQUENCE [like Current]
invariant
  cycle.has (Current)
  cycle.first = cycle.last
  ∀ i: 2 | .. | cycle.count ¦ cycle [i − 1].next = cycle [i]
end
```

Limitations

For any data structure, a static verifier might prefer a certain, distinct, but equivalent characterization of the chosen invariants.

All the class invariants must be complemented with specifications for the consistency of aggregate objects.

Thank you