Lecture 28

CPSC 110

Peyton Seigo

2018-11-14

Lecture 28

Accumulators

- Template tag: `(@template <TypeConsumed> accumulator)`
  - No `encapsulated` tag!

BinaryTree function template:

```
(@template BinaryTree accumulator)
(define (bst? bt0)
  ;; lower: Natural; lower bound of key at this node (based on parents)
  ;; upper: Natural; upper bound of key at this node (based on parents)
  (local [(define (fn-for-bst bt lower upper)
  (cond [(false? bt) (... lower ... upper)]
        [else
         (... lower
              upper
              (node-k bt)
              (node-v bt)
              (fn-for-bst (node-l bt) (... lower) (... upper))
              (fn-for-bst (node-r bt) (... lower) (... upper)))]))]

    (fn-for-bst bt0 ... ...)))
```

> **Tip**: Do NOT add multiple accumulators at once. Step through the process one accumulator at a time. This is important when we begin to add more than two accumulators in a function, especially for functions operating on graphs.

Finished function:

```
(@template BinaryTree accumulator)
(define (bst? bt0)
  ;; lower: Natural; lower bound of key at this node (based on parents)
  ;; upper: Natural; upper bound of key at this node (based on parents)
  (local [(define (fn-for-bst bt lower upper)
            (cond [(false? bt) true]
                  [else
                   (and (< lower (node-k bt) upper)
                        (fn-for-bst (node-l bt) lower (node-k bt))
                        (fn-for-bst (node-r bt) (node-k bt) upper))]))]
```

```

    (fn-for-bst bt0 0 +inf.0)))
```

Documentation on Numbers with regards to `+inf.0` (https://docs.racket-lang.org/reference/numbers.html). This value represents infinity and can be used with comparison operators.