Unstable-Funcons-beta: Memos *

The PLanCompS Project

`Memos.cbs` | PLAIN | PRETTY

Memos

[*Entity* memo-map
Funcon initialise-memos
Funcon memo-value
Funcon initialise-memo-value
Funcon memo-value-recall]

A memo is like a mutable variable, except that the memo is updated and accessed by a specified key, rather than by an allocated location. The collection of memos is represented by a mutable entity that maps keys to values.

Entity $\langle \_, \text{memo-map}(\_ : \text{maps}(\text{ground-values}, \text{values}))\rangle \longrightarrow$
$\langle \_, \text{memo-map}(\_ : \text{maps}(\text{ground-values}, \text{values}))\rangle$

Funcon $\text{initialise-memos}(\_ : \Rightarrow \text{values}) : \Rightarrow \text{values}$
Rule $\langle \text{initialise-memos}(X), \text{memo-map}(\_)\rangle \longrightarrow \langle X, \text{memo-map}(\text{map}(\ ))\rangle$

When key K is associated with value V, the funcon memo-value(K, X) simply gives V, without evaluating X. When K is not currently associated with any value, it associates K with the value computed by X.

Funcon $\text{memo-value}(K : \text{ground-values}, X : \Rightarrow \text{values}) : \Rightarrow \text{values}$

```
⇝ else(
      memo-value-recall(K),
      give(
        X,
        sequential(
          else(initialise-memo-value(K, given), null-value),
          memo-value-recall(K))))
```

The initialisation could fail due to memoisation of a (potentially different) value for K during the computation X. In that case, the value computed by X is simply discarded; a resource-safe funcon would take an extra argument to roll back the effects of X.

---

*Suggestions for improvement: `plancomps@gmail.com`.
Reports of issues: `https://github.com/plancomps/CBS-beta/issues`.

Funcon initialise-memo-value(_ : ground-values, _ : values) : ⇒ null-type

Rule
$$\frac{\text{map-unite}(M, \{K \mapsto V\}) \rightsquigarrow M'}{\langle \text{initialise-memo-value}(K : \text{ground-values}, V : \text{values}), \text{memo-map}(M)\rangle \longrightarrow \langle \text{null-value}, \text{memo-map}(M')\rangle}$$

Rule
$$\frac{\text{map-unite}(M, \{K \mapsto V\}) \rightsquigarrow (\ )}{\langle \text{initialise-memo-value}(K : \text{ground-values}, V : \text{values}), \text{memo-map}(M)\rangle \longrightarrow \langle \text{fail}, \text{memo-map}(M)\rangle}$$

Funcon memo-value-recall(_ : ground-values) : ⇒ values

Rule
$$\frac{\text{lookup}(M, K) \rightsquigarrow V}{\langle \text{memo-value-recall}(K : \text{ground-values}), \text{memo-map}(M)\rangle \longrightarrow \langle V, \text{memo-map}(M)\rangle}$$

Rule
$$\frac{\text{lookup}(M, K) \rightsquigarrow (\ )}{\langle \text{memo-value-recall}(K : \text{ground-values}), \text{memo-map}(M)\rangle \longrightarrow \langle \text{fail}, \text{memo-map}(M)\rangle}$$