Funktionale Analyse
zum
Problem des kürzeszen Wegs

Funktionale Analyse

Modularisierung:

- Suchfunktion
- Datenstruktur mit Zugriffsfunktionen
- Funktionen für Prioritätswarteschlange
- Funktionen zur Expansion des nächsten Schritts

Funktionale Analyse

Datenstruktur

- Kantenliste oder Knotenliste?
- Gewählt Knotenliste als Assoziationsliste

```
(define
  knoten
  '((A (B 69) (D 36) (M 36) (N 22))
    (B (A 69) (D 64) (Z 32) (H 30) (I 34) (X 26))
    (C (I 40) (M 31) (X 23))
    (D (A 36) (B 64) (L 95) (N 20))
… )
```

- Zugriff mit `(rest (assoc ))` gekapselt: hole-nachfolger

Funktionale Analyse

Funktionen für die Prioritätswarteschlange

- fuege-einen-ein
- fuege-alle-ein
- Hinweis:
  - Zugriffe allein funktional, keine eigene Datenstruktur

Funktionale Analyse

Weg expandieren

- alle-nachfolger liefert zu einem Weg alle fortsetzenden Wege, daher
- bildet die Hilfsfunktion neuer-weg zu einer fortsetzenden Kante den neuen Weg und
- filtert die Hilfsfunktion entferne-rueckwege alle die Wege, die zum vorigen Knoten zurückführen.
- Schlechtere Wege brauchen wegen der PrioWS nicht gefiltert zu werden.

Funktionale Analyse

Das eigentliche Suchprogramm

- stellt wegen der verallgemeinerten Funktionen für die Prioritätswarteschlange das Prädikat vor? bereit
- steuert die Rekursion und Aufrufe
- bekommt eine Aufrufhülle