Notizen zu Algorithmen II

Jens Ochsenmeier

15. Februar 2018

Inhaltsverzeichnis

1

Fortgeschrittene Datenstrukturen

Wir werden uns in diesem Kapitel mit Prioritätslisten beschäftigen. Es gibt noch viele weitere fortgeschrittene Datenstrukturen, z.B.

- monotone ganzzahlige Prioritätslisten (später im Kapitel "kürzeste Wege")
- perfektes Hashing
- Suchbäume mit fortgeschrittenen Operationen
- externe Prioritätslisten (später im Kapitel "Externe Algorithmen")
- Geometrische Datenstrukturen (siehe Kapitel "Geometrische Algorithmen")

1.1 Adressierbare Prioritätslisten

Eine adressierbare Prioritätsliste muss folgene Funktionen implementieren:

BUILD($\{e_1, \ldots, e_n\}$)	$M \coloneqq \{e_1, \ldots, e_n\}$
SIZE	**return** $\lvert M \rvert$
INSERT(e)	$M \coloneqq M \cup \{e\}$
MIN	**return** $\min M$
DELETEMIN	$e \coloneqq \min;\quad M \coloneqq M \setminus \{e\};\quad$ **return** e
REMOVE(h : Handle)	$e \coloneqq h;\quad M \coloneqq M \setminus \{e\};\quad$ **return** e
DECREASEKEY(h : Handle, k : Key)	key(h) $\coloneqq k$
MERGE(M')	$M \coloneqq M \cup M'$

Adressierbare Prioritätslisten haben viele Anwendungen, beispielsweise im Dijkstra-Algorithmus für kürzeste Wege oder in der Graphpartitionierung. Allgemein lassen sich adressierbare Prioritätslisten gut bei Greedy-Algorithmen verwenden, bei denen sich die Prioritäten (begrenzt) ändern.

Dijkstras Algorithmus

Berechne die Distanz zwischen einem Startknoten s und jedem anderen Knoten des Graphen.

```
DIJKSTRA(s : Node, T : Tree)
// Initialisieren: Distanz zu jedem Knoten ist ∞, zu Startknoten 0.
d = ⟨∞, ..., ∞⟩
d[s] = 0
// Startknoten zu PQ hinzufügen.
Q.insert(s)
```

- $d = \langle \infty, \cdots, \infty \rangle$
  Zu Beginn ist die Distanz zu jedem Knoten ∞.
- parent$[s] = s, d(s) = 0$
  Der Startknoten wird initialisiert.
- Q.insert(s)
  Prioritätsliste wird mit Startknoten initialisiert.