Algorithmen & Komplexität

Angelika Steger
Institut für Theoretische Informatik

steger@inf.ethz.ch

Kapitel 4:

Datenstrukturen

Datenstrukturen

Suchbäume:

- Operationen: *Find, Insert, Delete,*

Mengen:

- Operationen: *Find, Insert, Union*

Vorrangwarteschlangen:

- Operationen: *Insert, ExtractMin, DecreaseKey*

Wörterbücher:

- Operationen: *Find, Insert, Delete*

Kapitel 4:

Datenstrukturen

Kapitel 4.2:

Union-Find-Strukturen

Union-Find-Strukturen

Gegeben:

Datensätze partitioniert in (paarweise disjunkte) Mengen, wobei jede Menge durch einen in ihr enthaltenen Datensatz repräsentiert wird.

Gesucht:

Datenstruktur, die folgende Operationen unterstützt.

MakeNewSet(x)	→	Füge neuen Datensatz x ein; dieser bildet eine neue (einelementige) Menge.
Find(x)	→	Gebe Repräsentanten derjenigen Menge aus, die x enthält.
Union(x,y,r)	→	Vereinige die beiden Mengen zu denen x und y gehören; Repräsentant der neuen Menge sei r.

Anwendungsbeispiel: Algorithmus von Kruskal

Algorithmus 2.9 Algorithmus von Kruskal

Eingabe: Ein zusammenhängendes Netzwerk $N = (V, E, \ell)$

Ausgabe: [illegible]annbaum $T = (V, F)$ in N

{ *Initial* [illegible] }

Sortiere die [illegible], so dass $\ell(e_1) \leq \ldots \leq \ell(e_$

Setze $F := \emptyset$.

{ *Aufbau des Baums* }

for $i := 1$ **to** m **do**

  if $(V, F \cup \{e_i\})$ ist kreisfrei **then** $F := F \cup \{e_i\}$.

∀v ∈ V:
MakeNewSet(v)

Let e_i = {x,y}:
Union(x, y)

Let e_i = {x,y}:
if Find(x) ≠ Find(y) then …

Grundversion

Gegeben:

Datensätze partitioniert in (paarweise disjunkte) Mengen, wobei jede Menge durch einen in ihr enthaltenen Datensatz repräsentiert werde.

Idee:

Jede Menge wird durch einen zur Wurzel hin gerichteten Baum dargestellt (engl. *intree*). An jeder Wurzel ist gespeichert:

- Repräsentant der Menge
- Höhe des Baumes

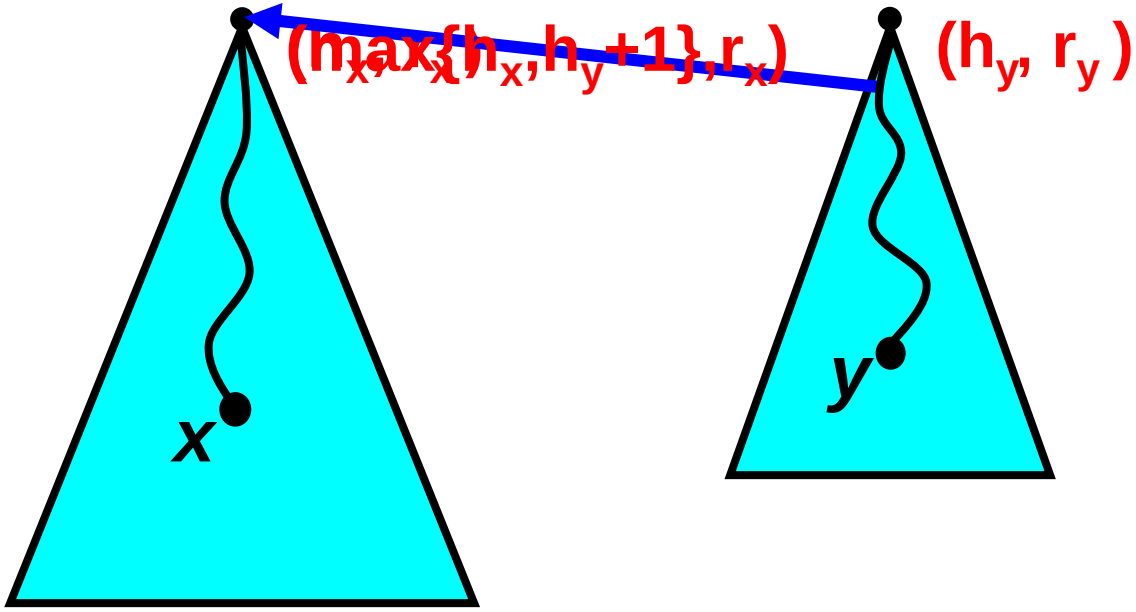
MakeNewSet(x) → x bildet einen Baum, der nur aus der Wurzel besteht; Höhe := 0, Repräsentant := x

Find(x) → Laufe von x zur Wurzel; gebe den an der Wurzel gespeicherten Repräsentanten aus,

Union(x,y,r) → Vereinige die Bäume …

Union(x,y)

Informationen an der Wurzel müssen angepasst werden.



$h_y \leq h_x$: Hänge Wurzel von y an die Wurzel von x an …

Eigenschaften

Lemma:
Für jeden in der Union-Find-Struktur enthaltenen Baum T gilt

$$\text{size}(T) \geq 2^{\text{height}(T)}$$

wobei $\text{size}(T)$ die Anzahl der Knoten in T und $\text{height}(T)$ die Höhe von T sei.

Korollar:
Für eine Union-Find Struktur mit n Elementen haben Find(x) und Union(x,y,r) Laufzeit **O(log(n)).**