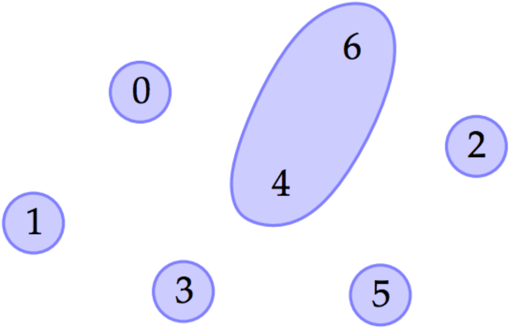
Tutorium 11

Algorithmen I SS 14

Institut für Theoretische Informatik

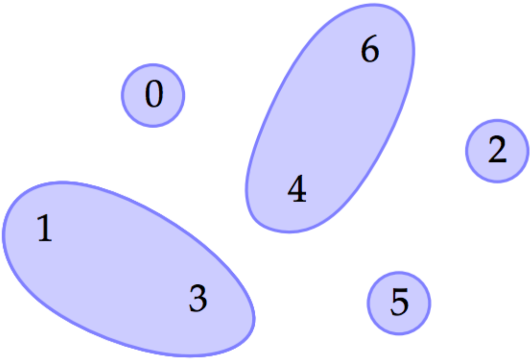
Beispiel Union-Find

union(4,6)
find(4) == find(6): true
find(1) == find(3): false
find(5) == find(2): false



Beispiel Union-Find

union(1,3)
find(4) == find(6): true
find(1) == find(3): true
find(5) == find(2): false



Optimierungen

Karlsruher Institut für Technologie

find() Alle traversierte Knoten direkt an die Wurzel gehängt:
$\rightarrow$ Reduziert Baumhöhe.

union() Der kleine Baum wird an den Größeren gehängt

Laufzeit

- Amortisierte Laufzeit pro Operation: $\mathcal{O}(\alpha(n))$
  - $\alpha(n)$ ist die Inverse Ackermannfunktion
  - sehr langsam wachsend:

$$\alpha(2^{2^{10^{19792}}}) < 5$$

Aufgabe: Bottleneck Shortest Path

Karlsruher Institut für Technologie

Sei $G = (V, E)$ ein zusammenhängender ungerichteter gewichteter Graph und $s, t \in V$. Ein kreisfreier Pfad P zwischen s und t heiße ein Bottleneck Shortest Path (BSP) für s und t, wenn das größte in P auftretende Kantengewicht minimal ist für alle Pfade zwischen s und t

1. Zeigen Sie: Ist T ein MST in G, dann ist der in T eindeutige Pfad P zwischen zwei Knoten $s, t \in V$ ein BSP in G für s und t.
2. Geben Sie einen Algorithmus an, der für gegebenen $G = (V, E)$, gegebene $s, t \in V$ und einen gegebenen MST T in G einen BSP P zwischen s und t ausgibt. Die Laufzeit soll dabei in $\mathcal{O}(|P|)$ liegen. Nehmen Sie an T liege in Form des Array *parent* vor.
3. Argumentieren Sie kurz warum ihr Algorithmus korrekt und die geforderte Laufzeit hat.

Kreativaufgabe: Streaming MST

Gegeben sei ein zusammenhängender Graph mit n Knoten und m Kanten. Die Knoten sind lokal gespeichert, während die Kanten über eine Netzwerkverbindung gestreamt werden. Sie können nicht lokal gespeichert werden, da nur $\mathcal{O}(n)$ Speicherplatz vorhanden ist.

Aufgabe 1: Gib einen Algorithmus an, der einen MST von G unter diesen Einschränkungen bestimmt.

Aufgabe 2: Verbessere diesen Algorithmus so, dass er nur $\mathcal{O}(m \log n)$ Rechenzeit benötigt.