Starker Zusammenhang

Wir können BFS oder DFS benutzen um Zusammenhangskomponenten in einem ungerichteten Graphen zu finden.

Die einzelnen Bäume des berechneten Waldes sind die Zhks.

Wie berechnen wir starke Zusammenhangskomponenten in einem gerichteten Graphen?

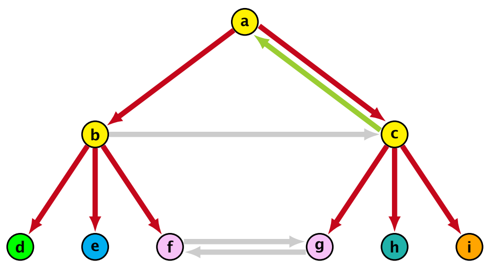
Starker Zusammenhang

Beobachtung
Die starken Zhks müssen innerhalb der durch die Tiefen- oder Breitensuche berechneten Bäume liegen, d.h. eine Zhk kann nicht Knoten von verschiedenen Bäumen enthalten.

Wir müssen nur die einzelnen Bäume in (starke) Zhks zerlegen.

Starker Zusammenhang

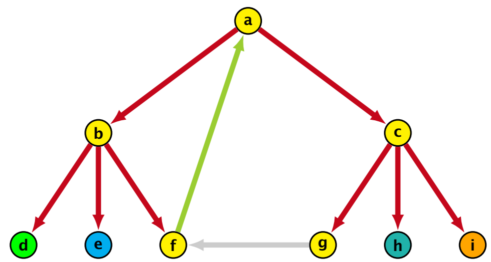
BFS-Baum:



Die (starken) Zhks sind nicht über Baumkanten verbunden.

Starker Zusammenhang

DFS-Baum:



In einem DFS-Baum sind die starken Zusammenhangskomponenten über die Baumkanten verbunden (wenn man diese als ungerichtet annimmt).

D.h. für 2 Knoten a und b, die in der gleichen Zhk sind, ist auch jeder Knoten auf dem (ungerichteten) Weg zwischen a und b im DFS-Baum in der gleichen Zhk.

Starker Zusammenhang

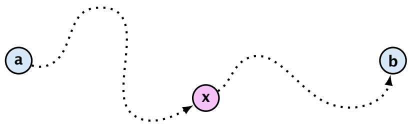
Beweis

Sei $a, b \in Z$, wobei Z starke Zhk ist. Wir zeigen zunächst $\mathrm{lca}(a,b) \in Z$ wobei $\mathrm{lca}(a,b)$, der kleinste gemeinsame Vorgänger von a und b im Baum ist.

Annahme:

a und b ist kleinstes Gegenbeispiel, d.h., Gegenbeispiel mit $\mathrm{dfsNum}(a) + \mathrm{dfsNum}(b)$ minimal. Sei $\mathrm{dfsNum}(a) < \mathrm{dfsNum}(b)$.

Wenn wir a besuchen ist b noch nicht besucht. Auf dem Pfad von a nach b muss es einen Knoten x geben, der entweder aktiv ist oder schon besucht wurde (sonst würde b im Teilbaum von a enden).



Starker Zusammenhang

- $x \in Z$, da er von a erreichbar ist und b erreichen kann.
- $\ell := \mathrm{lca}(a,x) \in Z$, da wir sonst ein kleineres Gegenbeispiel hätten.
- Falls $\mathrm{lca}(a,b)$ Vorgänger von ℓ folgt, dass ℓ und b ein kleineres Gegenbeispiel bilden (beachte $\ell \neq a$). ($\lightning$)
- Falls ℓ Vorgänger von $\mathrm{lca}(a,b)$ folgt, $\mathrm{lca}(a,b) \in Z$, da der Knoten von ℓ erreichbar ist und a erreichen kann. ($\lightning$)

Das zeigt dass $\mathrm{lca}(a,b) \in Z$. Damit gilt dies auch für Knoten zwischen $\mathrm{lca}(a,b)$ und a bzw. b, da diese von einem Knoten aus Z erreichbar sind, und einen Knoten aus Z erreichen können.

Starker Zusammenhang

Wir müssen nur Kanten aus dem DFS-Baum entfernen um die starken Zhks zu bestimmen. Kindknoten einer entfernten Knoten ist Wurzel von Zhk.

Welche Kanten werden entfernt?

Wie bestimmen wir Zhks?

- Immer wenn wir einen rekursiven Aufruf starten (einen Teilbaum betreten) legen wir den zugehörigen Knoten auf einen (separaten) Stack.
- Wenn wir einen Teilbaum (mit Wurzel v) verlassen und die zugehörige Kante, die in den Teilbaum führt entfernen wollen, gehören alle Knoten, die nach v auf dem Stack liegen zu Zhk. (nur dann werden sie entfernt).

Starker Zusammenhang

```
dfsTarjan()
    dfsCount = 1;
    foreach v ∈ V
        v->state  = initial;
        v->parent = NULL;
    foreach (s ∈ V)
        if s->state == initial
            dfsVisitTarjan(s)
```

Starker Zusammenhang

```
Input: directed graph G = (V,E);
Output: prints connected components of G

dfsVisitTarjan(v)
    v->state = active;
    v->num   = dfsCount++;
    S.push(v);
    foreach (x ∈ N[v])
        if (x->state == initial)
            x->parent = v;
            dfsVisitTarjan(x);
    if (cut parent edge of v)
        repeat
            x = S.pop();
            // add x to current component
        until (v == x)
        // print current component
```

Starker Zusammenhang

In der rekursiven Variante wird normalerweise beim Start eines rekursiven Aufrufs v auf den (impliziten) Stack gelegt, und bei Beendigung wieder entfernt.

Wir entfernen v jetzt nur wenn die gesamte zugehörige Zusammenhangskomponente entfernt wird. D.h. wenn wir bei einem Vorgänger von v die zugehörige Baumkante trennen.

Invariante/Ziel

Alle Knoten auf dem Stack können einen aktiven Knoten im Baum erreichen (deshalb ist die Zhk für diese Knoten noch nicht bestimmt).

Starker Zusammenhang

Um zu entscheiden ob wir eine Kante schneiden müssen, berechnen wir für jeden Knoten den Wert `v->low`.

`v->low`
Minimale dfsNum eines Knotens, der in der gleichen Zhk wie `v` liegt und über eine Folge von Baumkanten gefolgt von maximal einer Kreuzkante oder einer Rückwärtskante erreichbar ist.

Wenn wir das können, dann sind Knoten für die `v->low==v->num` ist, die Knoten an denen wir schneiden müssen.

Irgendwie scheint diese Definition nicht hilfreich, da wir die Zusammenhangskomponenten ja gerade berechnen wollen...

Starker Zusammenhang

```
Input: directed graph G = (V,E);
Output: prints connected components of G

dfsVisitTarjan(v)
    v->state = active;
    v->num   = v->low = dfsCount++;
    S.push(v); // setzt v->onStack
    foreach (x ∈ N[v])
        if (x->state == initial)
            x->parent = v;
            dfsVisitTarjan(x);
            v->low = min(v->low,x->low)
        else if (x->onStack)
            v->low = min(v->low,x->num)
    if (v->num == v->low)
        repeat
            x = S.pop(); // loescht x->onStack
            // add x to current component
        until (v == x)
        // print current component
```

Starker Zusammenhang