3.8 Laufzeit von BFS und DFS

Wenn man einen Graphen durchsucht, sollte man schon alle Knoten und alle Kanten ansehen; also lässt sich eine untere Schranke von $\Omega(n+m)$ nicht unterbieten.
Tatsächlich wird diese Schranke auch erreicht:

SATZ 3.13

Der Graphen-Scan-Algorithmus 3.7 lässt sich so implementieren, dass die Laufzeit $O(n+m)$ ist.

Beweis:

Wir nehmen an, dass G durch eine Adjazenzliste gegeben ist.

Für jeden Knoten verwenden wir einen Zeiger, der auf die „aktuelle" Kante für diesen Knoten in der Liste zeigt (d.h. auf den „aktuellen" Nachbarn).
Anfangs zeigt akt(x) auf das erste Element in der Liste.
In (2.3.1) wird der aktuelle Nachbar ausgewählt und der Zeiger weiterbewegt; wird das Listenende erreicht, wird x aus R entfernt und nicht mehr eingeführt.
Also ergibt sich eine Gesamtlaufzeit von $O(n+m)$. □

KOROLLAR 3.14

Mit Algorithmus 3.7 kann man alle Zusammenhangskomponenten eines Graphen berechnen.

Beweis:

Wende 3.7 an und überprüfe, ob $Y=V$ ist. Falls ja, ist der Graph zshgd.
Falls nein, haben wir eine Zusammenhangskomponente berechnet;
wir lassen den Algorithmus erneut für einen Knoten $s \in V \setminus Y$ laufen, usw.
Wieder wird keine Kante von einem Knoten aus doppelt angefasst,
also bleibt die Laufzeit linear, d.h. $O(n+m)$. □

Korollar 3.15
BFS und DFS haben Laufzeit $O(n+m)$.

Beweis:
Einfügen in R (Warteschlange bzw. Stapel) lässt sich jeweils in konstanter Zeit vornehmen, der Rest überträgt sich von Satz 3.13 □

3.9 Besondere Eigenschaften von BFS und DFS

Einfach gesagt:

- DFS ist eine bestmögliche, individuelle Suchstrategie mit lokaler Information.
- BFS ist eine bestmögliche, kooperative Suchstrategie mit globaler Information.

Konkret:

- DFS ist gut geeignet für die Suche nach einem Ausweg aus einem Labyrinth.
- BFS ist gut geeignet für die Suche nach kürzesten Wegen in einem ~~kürzesten~~

Satz 3.16 (Lokale Suche mit DFS)
DFS ist eine optimale lokale Suchstrategie in folgendem Sinne:

(1) DFS findet in jedem zusammenhängenden Graphen mit n Knoten einen Weg der Länge höchstens $2n-1$, der alle Knoten besucht.

(2) Für jede lokale Suchstrategie gibt es einen Graphen mit n Knoten, so dass der letzte Knoten erst nach einer Weglänge von $2n-1$ besucht wird.

Beweis: Übung!

Satz 3.18

(3) Am Ende ist für jeden erreichbaren Knoten $v \in Y$ die Länge eines kürzesten Weges von s nach v im Baum (Y,T) durch $\ell(v)$ gegeben.

(4) Am Ende ist für jeden erreichbaren Knoten $v \in Y$ die Länge eines kürzesten Weges von s nach v im Graphen (V,E) durch $\ell(v)$ gegeben.

Beweis:

(3) Sei $d_{(Y,T)}$ die Länge eines kürzesten Weges von s nach v in (Y,T).
Dann zeigen wir durch Induktion über $d_{(Y,T)}(s,v)$, dass für alle Knoten $d_{(Y,T)}(s,v) = \ell(v)$ gilt:

Induktionsanfang:

$d_{(Y,T)}(s,v) = 0$ gilt für $v=s$, und $\ell(s)=0$

Induktionsannahme:

Sei $d_{(Y,T)}(s,v) = \ell(v)$ für alle $v \in V$ mit $d_{(Y,T)}(s,v) \leq k-1$

Induktionsschritt:

sei $w \in V$ ein Knoten mit $d_{(Y,T)} = k$.

Dann gibt es im Baum (Y,T) einen eindeutigen Weg von s zu w.
Sei v der Vorgänger von w in diesem Weg, also $\{v,w\} \in T$.

Nach Induktionsannahme gilt

$$d_{(Y,T)}(s,v) = \ell(v),$$

außerdem ist

$$d_{(Y,T)}(s,w) = d_{(Y,T)}(s,v) + 1 \quad \text{(Abstand im Baum)}$$

und

$$\ell(w) = \ell(v) + 1. \quad \text{(wird in Alg 3.17 gesetzt)}$$

Also auch

$$d_{(Y,T)}(s,w) = \ell(w)$$

– und die Behauptung gilt.

(4) Wir brauchen zunächst eine Eigenschaft der Warteschlange R:

(*) Zu jedem Zeitpunkt gilt für die Warteschlange $R: v_i, v_{i+1}, \dots, v_k$, dass $\ell(v_i) \le \dots \le \ell(v_k) \le \ell(v_i)+1$.

Beweis durch Induktion über die Zahl z der aufgenommenen Kanten:

Induktionsanfang: $z=0$

Hier besteht die Warteschlange nur aus s, und (*) gilt.

Induktionsannahme:

Die Aussage gelte nach $z-1$ Kanten. Sei $R: v_i, v_{i+1}, \dots, v_k$ und

$$\ell(v_i) \le \ell(v_{i+1}) \le \dots \le \ell(v_k) \le \ell(v_i)+1$$

Induktionsschritt:

Wir fügen eine weitere Kante ein. Werden dafür zunächst Knoten aus R gelöscht (weil sie keine neuen Nachbarn haben), ändert das nichts an (*). Sei v_j danach der erste Knoten, für den eine neue Kante $e = \{v_j, v_{k+1}\}$ eingefügt wird. Dann bekommt man

$$R: v_j, v_{j+1}, \dots, v_k, v_{k+1}$$

mit $\ell(v_i) \le \dots \le \ell(v_j) \le \dots \le \ell(v_k) \le \ell(v_i)+1$

Wegen $\ell(v_i) \le \ell(v_j)$ gilt auch $\ell(v_i)+1 \le \ell(v_j)+1 = \ell(v_{k+1})$.

Damit erhält man

$$\ell(v_j) \le \dots \le \ell(v_k) \le \ell(v_i)+1 \le \ell(v_j)+1 = \ell(v_{k+1}),$$

und (*) gilt weiterhin.

Jetzt nehmen wir an, dass es am Ende des Algorithmus 3.17 einen Knoten $w \in V$ gibt, für den nicht ein kürzester Weg gefunden wurde, also

$$d(s,w) < d_{(Y,T)}(s,w) = \ell(w).$$

Unter den Knoten mit dieser Eigenschaft wählen wir einen mit minimalem Abstand von s in G.

Sei P ein kürzester s-w-Weg in G, und sei $e = \{v,w\}$ die letzte Kante in P, d.h.

$$d(s,w) = d(s,v) + 1.$$

Wir haben $d(s,v) = d_{(Y,T)}(s,v)$,

aber $d(s,w) < d_{(Y,T)}(s,w)$,

also gehört e nicht zu T.

Außerdem ist

$$\ell(w) = d_{(Y,T)}(s,w) > d(s,w) = d(s,v)+1 = d_{(Y,T)}(s,v)+1 = \ell(v)+1.$$

Wäre $w \in R$, hätten wir wegen

$$\ell(w) > \ell(v)+1$$

also einen Widerspruch zu Eigenschaft $(*)$.

Also wäre $w \notin Y$. Dann wäre die Kante $e = \{v,w\}$ zum Zeitpunkt der Entfernung von v aus R aber eine Verbindung von v mit einem Knoten $v \notin Y$, im Widerspruch zur Abfrage in .

□