Prof. Dr. Sándor Fekete
Nils Schweer

Algorithmen und Datenstrukturen
Übung 2 vom 18.11.2009

Abgabe der Lösungen am Mittwoch, den 02.12.09, vor der Abteilung *Algorithmik*.
Bitte die Blätter vorne deutlich mit eigenem **Namen** und **Gruppennummer** versehen!

Aufgabe 1 (Breitensuche und Tiefensuche):

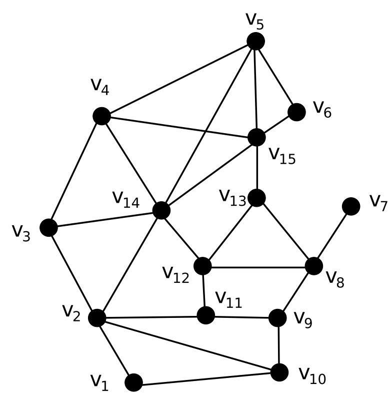

Abbildung 1: Der Graph G.

a) Wende Breitensuche mit Startknoten v_1 auf den Graphen G in Abbildung 1 an.

b) Wende Tiefensuche mit Startknoten v_1 auf den Graphen G in Abbildung 1 an.

(Kommt zu einem Zeitpunkt mehr als ein Knoten für den nächsten Schritt in Frage, wähle denjenigen mit kleinstem Index. Gib nach jeder Veränderung den Stack bzw. die Queue an und zeichne am Ende den gefundenen Baum.)

(10+10 Punkte)

Aufgabe 2 (Breitensuche):

Sei $G = (V, E)$ ein Graph und $s \in V$ ein Knoten; für jeden beliebigen Knoten $x \in V$ bezeichne $d(s, x)$ die Länge eines kürzesten Weges von s nach x. Sei $e = \{u, v\} \in E$ eine Kante.

a) Zeige: $d(s, v) \leq d(s, u) + 1$.

b) Zeige oder widerlege: $d(s, u) \leq d(s, v) + 1$.

c) Muss immer $d(s, v) = d(s, u) + 1$ oder $d(s, u) = d(s, v) + 1$ gelten?

(7+6+7 Punkte)

Aufgabe 3 (Tiefensuche in Labyrinthen):

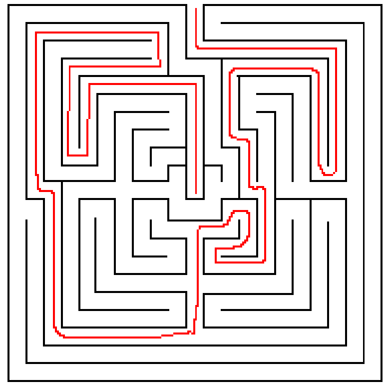

Abbildung 2: Ein Labyrinth.

Um einen Ausweg aus einem Labyrinth (vgl. Abbildung 2) zu finden, kann man sich der Tiefensuche bedienen. In dieser Aufgabe soll bewiesen werden, dass Tiefensuche dabei in einem gewissen Sinne eine bestmögliche Strategie ist.
Modelliert man die Kreuzungen in einem Labyrinth als Knoten und die Verbindungen zwischen zwei Kreuzungen als Kanten, ist die Aufgabe, von einem gegebenen Startknoten s (aktuelle Position) einen anderen Knoten t (den Ausgang) in einem Graphen G zu erreichen. Erschwerend kommt hinzu, dass man keine globale Sicht auf das Labyrinth hat, sondern nur eine lokale, d.h. an einer Kreuzung erkennt man nur die abgehenden Verbindungen nicht aber deren Ziele; im Modell muss man also eine Kante wählen, ohne dabei den anderen Endknoten zu kennen.
Dabei kostet das Durchlaufen einer Kante jeweils einen Schritt.

a) Zeige, dass bei einer Tiefensuche keine Kante mehr als zweimal durchlaufen wird. Folgere, dass für einen Graphen mit insgesamt $n+1$ Knoten (einschließlich Startknoten) der Ausgang t in höchstens $2n-1$ Schritten gefunden wird.

Damit hat man gezeigt, dass man mit Hilfe der Tiefensuche *immer* nach $2n-1$ Schritten den Ausgang gefunden hat. Jetzt wollen wir uns überlegen, dass es keine Strategie geben kann, die für jedes Labyrinth mit $n+1$ Kreuzungen weniger als $2n-1$ Schritte benötigt, d.h. für jede Strategie gibt es ein Labyrinth, für das das Finden des Ausganges mindestens $2n-1$ Schritte benötigt. Diesen Sachverhalt soll man sich zunächst an einem Beispielgraphen verdeutlichen (Teil b)) und dann allgemein beweisen (Teil c)).

b) Gib einen Graphen mit $n+1=6$ Knoten, so dass für jeden gewählten Suchpfad einer der Knoten erst nach $2n-1=9$ Schritten erreicht wird.

   (Man kann dabei davon ausgehen, dass aufgrund der lokalen Sichtweise im Zweifelsfall die “falsche” Entscheidung getroffen wird. Insbesondere ist also die Strategie “Gehe vom Startknoten zum Zielknoten” für alle Graphen, die keine Pfade sind nicht realisierbar / zulässig. Durch “Sackgassen” kann man die Strategie zwingen, unnötige Arbeit zu tun. Ebenso darf man den Knoten, der durch den Suchpfad zuletzt besucht wird, zum Ausgang erklären.)

c) Argumentiere, dass $2n - 1$ Schritte in gewissem Sinne optimal sind: Gib dazu eine Menge von Graphen an, so dass für jeden gewählten Suchpfad einer der Knoten erst nach $2n - 1$ Schritten erreicht wird.

(Es gelten die gleichen Hinweise wie in b). Es genügt, die Menge von Graphen mit einer Abbildung und Worten zu beschreiben.)

(10+3+7 Punkte)