Technische Universität Chemnitz
Professur Theoretische Informatik
Andreas Goerdt, Frank Schädlich

Chemnitz, den 21. Februar 2002

Klausur Theoretische Informatik I

Alle Studiengänge

Wintersemester 2001/02

Hinweise

Die Aufgaben 1-7 sind von allen Klausurteilnehmern zu bearbeiten. Prüfungsteilnehmer des Studienganges Mechatronik haben zusätzlich die Aufgaben 8 und 9 zu lösen. Die Klausur ist bestanden, wenn mindestens 50% aller möglichen Punkte erzielt werden.

Es sind keine Hilfsmittel zugelassen. Alle Blätter sind zu numerieren sowie unbedingt mit Namen und Matrikelnummer zu versehen. Bitte vermerken Sie die Anzahl Ihrer Lösungsblätter auf der ersten Seite und verwenden Sie nur radierfeste Stifte in den Farben blau oder schwarz. Bitte fangen Sie mit einer neuen Aufgabe ein neues Lösungsblatt an.

Aufgabe 1: **2 + 2 + 2 Punkte**

Es sei $G = (V,\ E)$ ein *ungerichteter* und *zusammenhängender* Graph. Auf G wird die Tiefensuche durchgeführt. Dabei werden die Entdeckzeitpunkte der Knoten im Feld $d = d[1 \ldots |V|]$ gespeichert. Während der Ausführung der Prozedur DFS-Visit(u) wird die Adjazenzliste von u durchlaufen. Sei v ein beliebiger Knoten, der in dieser Adjazenzliste gefunden wird. Die nachfolgenden Fragen beziehen sich auf den Zeitpunkt, zu dem v auf der Adjazenliste von u gefunden wird. Geben sie eine kurze Begründung zu Ihrer Antwort an.

a) Ist v immer weiß oder grau gefärbt?

b) Gilt immer $d[v] < d[u]$, wenn v grau gefärbt ist?

c) Ist die Kante $\{u, v\}$ immer eine Rückwärtskante der Suche, wenn v grau gefärbt ist?

Aufgabe 2: **3 + 3 Punkte**

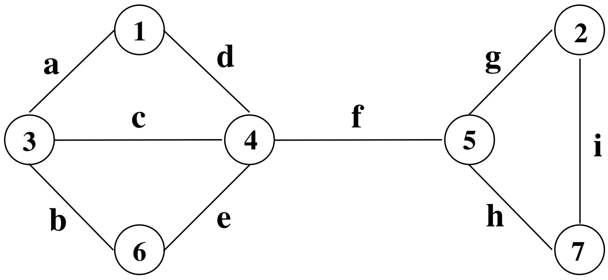
Auf der Menge der natürlichen Zahlen $\mathbb{N} = \{0, 1, 2, \ldots\}$ sei nur der Vergleich ($<$) als Verknüpfung zugelassen. Wir betrachten hier die Datenstruktur des Heaps der Vorlesung. Als Laufzeit einer Operation wird die Anzahl der durchgeführten Vergleiche gezählt.

a) Geben Sie die optimale Laufzeit an, in der ein Heap aus n beliebigen natürlichen Zahlen aufgebaut werden kann. Demonstrieren Sie das Verfahren mit optimaler Laufzeit an dem Array $A = [7, 6, 5, 4, 3, 2, 1]$ auf dem Lösungsblatt. Tragen Sie hierzu drei signifikante Zwischenschritte ein. Das Minimum soll an die Wurzel kommen.

b) Die Operation DeleteMin angewandt auf einen Heap löscht das Minimum und stellt wieder einen Heap her. Ist es möglich, DeleteMin() so zu implementieren, daß eine n-malige Ausführung der Operation eine Zeit von $O(n)$ benötigt? Begründen Sie kurz Ihre Antwort.

Aufgabe 3: **3 + 3 Punkte**

Gegeben sei der folgende ungerichtete Graph $G = (V, E)$ mit $V = \{1, \ldots, 7\}$ und $E = \{a, \ldots, i\}$:



Ausgehend von einer Adjazenzlistendarstellung mit aufsteigender Sortierung der Knoten in den Adjazenzlisten soll der Algorithmus zur Berechnung des Low-Arrays ablaufen. Er soll bei Knoten 1 beginnen.

a) Geben Sie für alle Knoten die Entdeckzeit d, die Beendezeit f und den Low-Wert an, wie sie sich aus dem Algorithmus ergeben. (Es genügt hierzu, auf dem Lösungsblatt jeden Knoten in der Form Entdeckzeit/Beendezeit/Low-Wert zu kennzeichnen.)

b) Von diesen Werten ausgehend laufe der Algorithmus zur Berechnung zweifacher Zusammenhangskomponenten auf dem Graphen ab. Geben Sie den Stackinhalt in jedem Schritt und die Ausgaben des Algorithmus an.

Aufgabe 4: **4 Punkte**

Betrachten Sie die Rekurrenz

$$\begin{aligned} T(1) &= T(2) = c \\ T(n) &= T\left(\frac{n}{2}\right) + T\left(\frac{n}{4}\right) + n. \end{aligned}$$

Dabei ist n eine Zweierpotenz und c eine Konstante mit $c > 0$. Geben Sie die Größenordnung von $T(n)$ möglichst genau (in der „O-“ bzw. Landaunotation) an und begründen Sie kurz Ihre Antwort.

Aufgabe 5: **4 + 5 Punkte**

Betrachten Sie die Rekurrenz

$$\begin{aligned} T(1) &= 1 \\ T(n) &= 3 \cdot T\left(\frac{n}{2}\right) + n, \end{aligned}$$

wobei n wieder eine Zweierpotenz ist.

a) Leiten Sie eine genaue explizite Formel für $T(n)$ her. Hinweis: Es bietet sich an, den Aufrufbaum zu verwenden.

b) Beweisen Sie durch Induktion, daß für $T(n)$ die Formel aus a) gilt.

Aufgabe 6: **5 Punkte**

Transformieren Sie

$$F = (a \wedge \neg b) \vee (\neg(c \vee d))$$

in eine erfüllbarkeitsäquivalente Formel in 3-KNF unter Verwendung des Polynomialzeitverfahrens der Vorlesung.

Aufgabe 7: **2 + 5 + 4 Punkte**

Es seien $v = a_1 \ldots a_m$ und $w = b_1 \ldots b_n$ Worte über einem gegebenen Alphabet. Es sei

$$\text{kgO}(v, w) = \text{die Länge einer kürzesten gemeinsamen Oberfolge von } v \text{ und } w.$$

Dabei ist ein Wort u Oberfolge von v genau dann, wenn v eine Teilfolge (im Sinne der Vorlesung) von u ist.

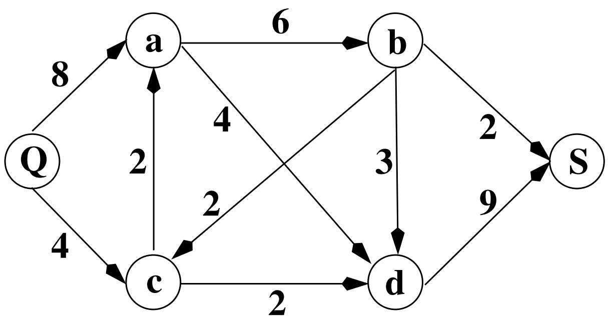
a) Geben Sie $\text{kgO}(ab, bac)$ und $\text{kgO}(aa, aba)$ an.

b) Geben Sie ein rekursives Gleichungssystem zur Berechnung von $\text{kgO}(v, w)$ an.

c) Demonstrieren Sie die Arbeitsweise eines auf dem Prinzip des dynamischen Programmierens basierenden Algorithmus, der bei $v = a_1 \ldots a_m$ und $w = b_1 \ldots b_n$ die Zeit $O(n \cdot m)$ benötigt. Füllen Sie die Tabelle auf dem Lösungsblatt für $v = abbc$ und $w = abcb$ entsprechend aus. Geben Sie die dadurch gefundene Lösung zusätzlich an.

Aufgabe 8 (nur Studiengang Mechatronik): **2 + 2 + 2 Punkte**

a) Bestimmen Sie den maximalen Fluß im folgenden Netzwerk mit der Quelle Q und der Senke S. Es genügt, auf dem Lösungsblatt den Fluß jeder Kante einzutragen.



b) Geben Sie zu dem Netzwerk aus a) und dem gefundenen maximalen Fluß den Restgraphen an und lesen Sie daraus die Lösung für das MinCut-Problem auf diesem Graphen ab.

c) Es sei $G = (V, E, c)$ mit $c\colon V \times V \to \mathbb{N}$ und $\max_{e \in V \times V} c(e) = k$ ein Flußnetzwerk. Geben Sie ein Beispiel an, bei dem der Algorithmus von Ford-Fulkerson eine Laufzeit von $\Omega(k)$ hat.

Aufgabe 9 (nur Studiengang Mechatronik): **2 + 2 + 2 Punkte**

Es sei $G = (V,\ E)$ ein *ungerichteter* und *zusammenhängender* Graph. Auf G wird die Breitensuche durchgeführt. Bei der Bearbeitung eines Knotens u wird seine Adjazenzliste durchlaufen. Es werde v in dieser Adjazenzliste gefunden. Die nachfolgenden Aussagen beziehen sich auf den Zeitpunkt, zu dem v auf der Adjazenliste von u gefunden wird. Welche Aussagen sind wahr? Beachten sie, daß $d = d[1 \ldots |V|]$ das Distanzarray bzw. $\pi = \pi[1 \ldots |V|]$ das Vorgängerarray ist. Begründen Sie kurz Ihre Antwort.

a) Der Knoten v ist nie schwarz gefärbt.

b) Wenn $d[v] = d[u] + 1$ gilt, dann ist $\pi[v] = u$.

c) Es ist $d[u] - 1 \leq d[v] \leq d[u] + 1$.