

Prof. Petra Mutzel  
Günther Raidl  
Ivana Ljubić  
René Weisskircher

Sommersemester 2000/2001

**Prüfung zur Vorlesung  
Algorithmen und Datenstrukturen 2  
19. Juni 2001**

1. Machen Sie bitte die folgenden Angaben in deutlicher Blockschrift:

Name: \_\_\_\_\_ Vorname: \_\_\_\_\_

Matrikelnummer: \_\_\_\_\_ Studienkennzahl: \_\_\_\_\_

2. Legen Sie während der Vorlesungsprüfung Ihren Studentenausweis vor sich auf das Pult.
3. Schreiben Sie die Lösungen direkt auf das jeweilige Aufgabenblatt. Wenn Ihnen das Papier ausgeht, bitten Sie die Aufsicht um Nachschub. Es ist nicht erlaubt, eigenes Papier zu verwenden!
4. Denken Sie daran, dass keinerlei Hilfsmittel erlaubt sind (weder Taschenrechner, irgendwelche Unterlagen, Handys,...).

**VOR DER ABGABE AUSZUFÜLLEN:**

5. Geben Sie bitte die Anzahl der zusätzlich abgegebenen Blätter an: \_\_\_\_\_
6. Kreuzen Sie bitte die von Ihnen bearbeiteten Aufgaben in der ersten Zeile der Tabelle an:

Aufgabe	A 1	A 2	A 3	A 4	A 5	A 6	$\Sigma$	Note
bearbeitet							—	—
maximale Punktzahl	10	15	6	10	6	13	60	—
erreichte Punktzahl								

Viel Erfolg!

**Aufgabe 1: Depth-First Search****10 Punkte**

Gegeben sei ein ungerichteter Graph  $G = (V, E)$ . Geben Sie eine nicht-rekursive Implementierung des Depth-First Search Algorithmus an.

(Hinweis: um eine richtige Implementierung zu entwickeln, sollten/dürfen Sie die Stack Datenstruktur benutzen.)

**Aufgabe 2: Maximaler Wald****15 Punkte**

Ein Wald ist ein ungerichteter Graph ohne Kreise. Ein maximaler Wald des gewichteten Graphen  $G = (V, E)$  mit  $w : E \mapsto \mathbb{Q}$  ist eine Menge  $W \subseteq E$ , die ein Wald ist, und die die folgende Summe maximiert:

$$\sum_{e \in W} w(e).$$

Verwenden Sie die Idee von Kruskal's Greedy Algorithmus, um den maximalen Wald für das folgende Beispiel zu finden.

- Schreiben Sie einen Pseudo-Code dafür.
- Zeichnen Sie den Graphen für jeden Schritt Ihres Algorithmus.

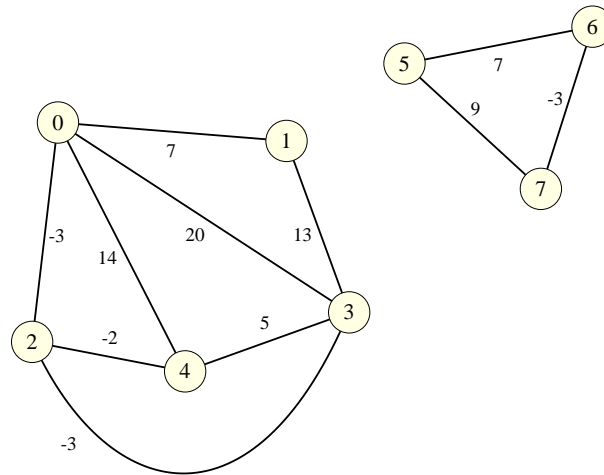


Abbildung 1: Maximaler Wald - Beispiel

**Aufgabe 3: WhatAmI****6 Punkte**

Gegeben sei folgender Algorithmus für das symmetrische TSP.

---

**input:** Vollständiger ungerichteter Graph  $G = (V, E)$  ( $|V| \geq 3$ ) mit den Gewichten  $l : V \times V \mapsto \mathbb{R}^+$ .**output:** Rundtour  $K$  durch alle Knoten, die jeden Knoten genau einmal besucht.**ALGORITHM**( $G, K$ )Wähle Anfangskreis  $K$  mit 3 Knoten  $v_1, v_2, v_3$  und setze  $W = V \setminus \{v_1, v_2, v_3\}$ .**while**  $W \neq \emptyset$  **do**    Wähle den Knoten  $a \in W$  mit dem kürzesten Abstand von einem Knoten  $b$  des Kreises  $K$ ,  
    d.h.

$$l(a, b) = \min\{l(i, j) \mid i \in W, j \in V \setminus W\};$$

 $W = W \setminus \{a\};$     Entferne eine Kante  $(b, c)$  des Kreises  $K$ , und füge die Kanten  $(a, b)$  und  $(a, c)$  so ein, dass

$$\min\{l(a, b) + l(a, i) - l(b, i) \mid i \in K\}$$

erreicht wird, d. h. :

 $K = K \setminus \{(b, c)\} \cup \{(a, b)\} \cup \{(a, c)\};$ **end while****return**  $K$ ;

---

Es handelt sich hierbei um:



einen Dynamischen Programmierungsalgorithmus.



einen exakten Algorithmus.



eine Verbesserungsheuristik.



eine Konstruktionsheuristik.

(Bitte kreuzen Sie richtigen Antworten an!)

#### Aufgabe 4: Euklidisches TSP

10 Punkte

Wenden Sie den in der Aufgabe 3 gegebenen Algorithmus auf das folgende Beispiel für das Euklidische TSP. Veranschaulichen Sie **alle** Schritte, die zur Konstruktion einer Lösung führen. Der startende Kreis  $K$  ist gegeben.

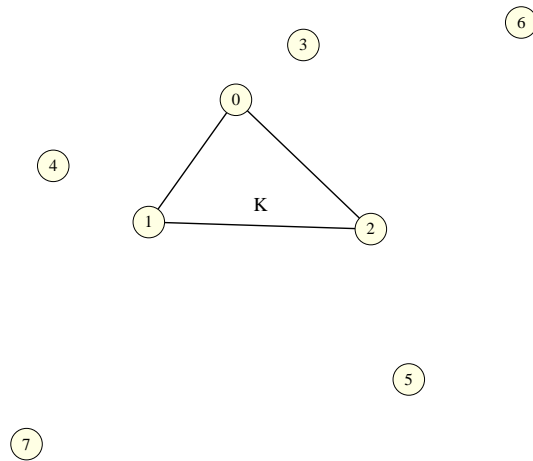
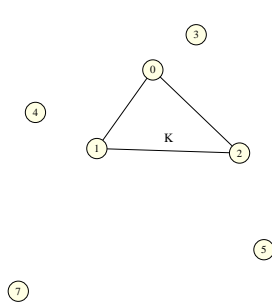
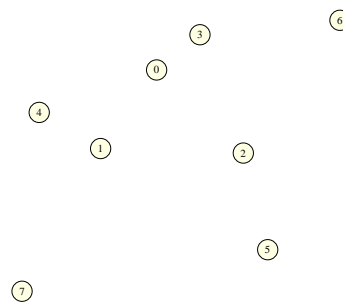


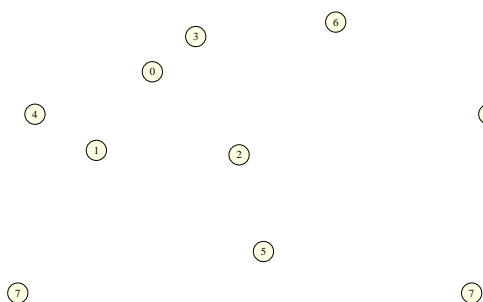
Abbildung 2: Euklidisches TSP - Beispiel



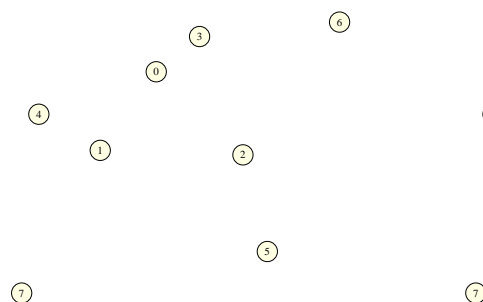
(a) Schritt 1



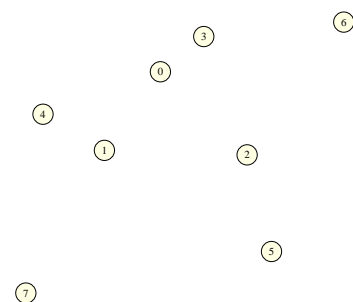
(b) Schritt 2



(c) Schritt 3



(d) Schritt 4



(e) Schritt 5

**Aufgabe 5: Evolutionäre Algorithmen****6 Punkte**

Wir möchten die Bewertungen  $f(x_i)$  der Lösungen  $x_i$  in der Population ( $i = 1, \dots, N$ ) über eine lineare Funktion  $F(f(x_i)) = a \cdot f(x_i) + b$  skalieren. Jede Lösung wird dann proportional zu diesen skalierten Bewertungen selektiert. Dabei wollen wir erreichen, dass die durchschnittliche Bewertung  $\bar{f} = \frac{\sum_{i=1}^N f(x_i)}{N}$  nicht verändert wird (d.h.  $F(\bar{f}) = \bar{f}$ ), aber zu erwarten ist dass die schlechteste Lösung mit der Bewertung  $f_{min}$  10 mal so selten wie der Durchschnitt selektiert wird (d.h.  $F(f_{min}) = \frac{\bar{f}}{10}$ ).

Wie sieht die Skalierungsfunktion  $F$  aus?

☐  $F(x) = \frac{9\bar{f}}{10(\bar{f}-f_{min})} \cdot x + \frac{10\bar{f}f_{min}-\bar{f}^2}{10(f_{min}-\bar{f})}$

☐  $F(x) = \frac{9\bar{f}}{10(\bar{f}-f_{min})} \cdot x + \frac{\bar{f}^2-10\bar{f}f_{min}}{10(\bar{f}-f_{min})}$

☐  $F(x) = \frac{\bar{f}}{10(\bar{f}-f_{min})} \cdot x + \frac{\bar{f}^2-10\bar{f}f_{min}}{10(\bar{f}-f_{min})}$

☐  $F(x) = \frac{10\bar{f}}{9(\bar{f}-f_{min})} \cdot x + \frac{\bar{f}^2-10\bar{f}f_{min}}{10(\bar{f}-f_{min})}$

**Aufgabe 6: Scan-line Prinzip****13 Punkte**

Eine Menge horizontaler Liniensegmente  $S = \{s_1, \dots, s_n\}$  in der Ebene, die durch die Anfangs- und Endpunkte  $(s_i^a, s_i^e, \forall i = 1, \dots, n)$  repräsentiert sind, ist gegeben. Liniensegment  $s$  “bedeckt” Liniensegment  $t$  wenn

$$s^a \leq t^a < t^e \leq s^e.$$

- a.) Geben Sie einen möglichst effizienten Pseudocode an, der alle Liniensegmente, die mit mindestens 2 anderen Liniensegmenten bedeckt sind, berechnet.
- b.) Geben Sie die Laufzeit Ihres Algorithmus für den Worst-Case-Fall in  $\Theta$ -Notation an.