Lehrstuhl für Informatik I
Algorithmen und Komplexität
Universität Würzburg

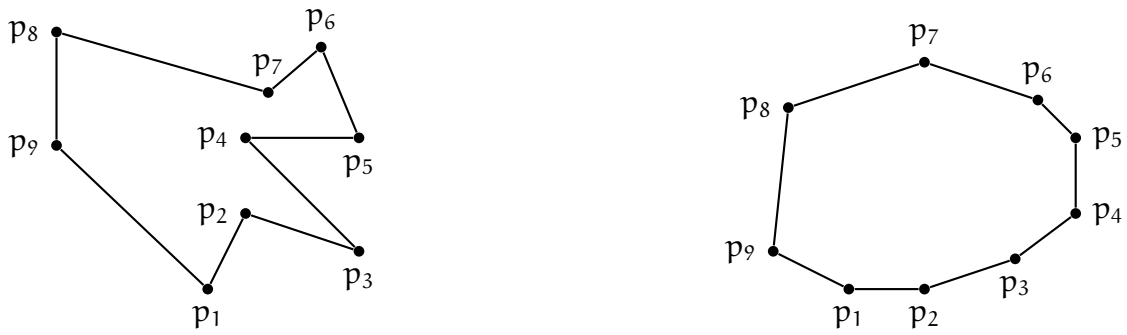
Würzburg, den 6. Mai 2026

Prof. Dr. Alexander Wolff
Antonio Lauerbach, M. Sc.

3. Übungsblatt zur Vorlesung Algorithmische Graphentheorie (Sommer 2026)

Aufgabe 1 – Triangulierungen und Dynamische Programmierung

Ein *Polygon* ist ein geschlossener Streckenzug in der Ebene – eine Folge von n verbundenen Punkten $p_1, p_2, \ldots, p_n$, die wieder am Startpunkt p_1 endet. Die Punkte $p_1, p_2, \ldots, p_n$ des Polygons werden *Ecken* genannt und die Strecken zwischen den Ecken *Polygonkanten*. Ein Polygon ist *konvex*, wenn jede Strecke mit Start- und Endpunkt im Polygon auch selbst vollständig im Polygon enthalten ist; siehe Abbildung 1.



(A) nicht-konvexes Polygon

(B) konvexes Polygon

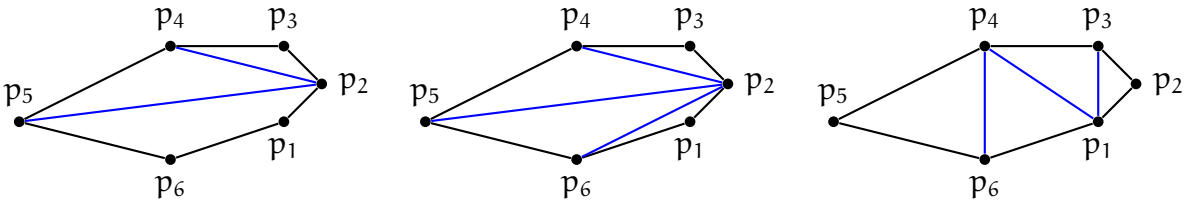
ABBILDUNG 1: Beispiele für Polygone.

Eine *Triangulierung* eines Polygons ist eine nicht-erweiterbare Menge von Strecken, die nicht-benachbarte Ecken verbinden, im Inneren des Polygons liegen, keine Polygonkanten schneiden und sich nicht gegenseitig schneiden. Wir nennen diese Strecken *Diagonalen*.

Wenn wir ein konvexes Polygon triangulieren, so müssen wir offensichtlich nur darauf achten, dass sich die Diagonalen nicht gegenseitig schneiden. Die *Kosten* einer Triangulierung sind die Summe der euklidischen Längen aller ihrer Diagonalen. Entsprechend heißt die Triangulierung mit den geringsten Kosten *kostenminimal*; siehe Abbildung 2.

Wir entwickeln nun schrittweise ein dynamisches Programm, um die Kosten einer kostenminimale Triangulierung eines gegebenen konvexen Polygons P zu bestimmen.

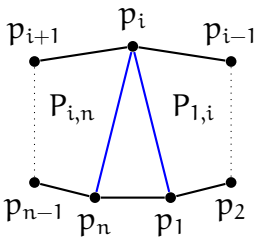
Sie können annehmen, dass wir die Länge $d(p_i, p_j)$ einer Diagonalen (p_i, p_j) in konstanter Zeit bestimmen können und dass es keine drei Ecken des Polygons gibt, die auf einer Geraden liegen.



(A) keine Triangulierung (B) Triangulierung, die nicht kostenminimal ist (C) kostenminimale Triangulierung

ABBILDUNG 2: Triangulierungen eines konvexen Polygons. Die Diagonalen sind blau gefärbt.

a) In einer kostenminimalen Triangulierung von P ist die Polygonkante (p_1, p_n) Teil eines Dreiecks $\Delta p_1 p_i p_n$, wobei $i \in \{2, \ldots, n-1\}$. Das Dreieck $\Delta p_1 p_i p_n$ trennt von P zwei (konvexe) Teilpolygone $P_{1,i}$ und $P_{i,n}$ ab.
Angenommen wir würden die Kosten für eine kostenminimale Triangulierung von $P_{1,i}$ und $P_{i,n}$ kennen, wie könnten wir daraus die Kosten einer kostenminimalen Triangulierung von P berechnen?
Beachten Sie auch die Spezialfälle $i = 2$ und $i = n - 1$. **2 Punkte**



b) Beschreiben Sie ein dynamisches Programm in Worten, welches für ein gegebenes konvexes Polygon die Kosten einer kostenminimalen Triangulierung berechnet.

Das dynamische Programm soll polynomielle Laufzeit haben und eine Tabelle $A[\cdot, \cdot]$ verwenden, welche für jedes Teilpolygon $P_{i,j}$ von P mit Ecken $p_i, p_{i+1}, \ldots, p_{j-1}, p_j$ und $1 \leq i < j \leq n$ einen Eintrag $A[i, j]$ für die Kosten einer kostenminimalen Triangulierung enthält. **4 Punkte**

c) Geben Sie scharfe obere Schranken für die asymptotische Laufzeit und den asymptotischen Speicherbedarf Ihres dynamischen Programms an. **2 Punkte**

Aufgabe 2 – TSP mit Wiederholungen

In der Vorlesung haben wir gesehen, dass sich das Metrische TSP-Problem approximieren lässt, während das allgemeine TSP-Problem nicht approximierbar ist (falls P≠NP). In dieser Aufgabe betrachten wir eine andere Variante von TSP, die ebenfalls eine Approximation zulässt. Gegeben sei ein ungerichteter vollständiger Graph G mit Kantenkosten $c\colon E(G) \to \mathbb{R}_{\geq 0}$. Gesucht ist eine Rundreise K in G, die jeden Knoten *mindestens* einmal besucht und minimale Kosten $\sum_{e \in K} c(e)$ hat. Im Gegensatz zu TSP dürfen hier also Knoten und Kanten mehrfach besucht werden.

a) Reduzieren Sie das Problem auf Metrisches TSP. **3 Punkte**

b) Benutzen Sie Teilaufgabe a), um eine 2-Approximation für das Problem zu beschreiben. **3 Punkte**

Aufgabe 3 – Metrisches TSP

In der Vorlesung wurde ein Approximationsalgorithmus auf Basis von Spannbäumen für das metrische TSP vorgestellt. Dieser besitzt *Güte* 2, d. h. die Rundreise, die der Algorithmus liefert, ist höchstens doppelt so lang wie eine optimale Rundreise.

Sei G ein vollständiger, metrischer, ungerichteter Graph mit Kantengewichten. Betrachten Sie den folgenden Algorithmus:

CompleteHamilton(Graph G, Kostenfunktion $c\colon E(G) \to \mathbb{R}_{\geq 0}$)

wähle $s \in V(G)$ beliebig
wähle eine Kante $\{s,t\} \in E(G)$ mit $c(\{s,t\})$ minimal
$C \leftarrow \langle \{s,t\}, \{t,s\} \rangle$
while $V(C) \neq V(G)$ **do**
  wähle einen Knoten $v \in V(G) \setminus V(C)$, dessen Abstand zu C am geringsten ist
  sei w ein Knoten in C mit $c(\{v,w\})$ minimal
  sei u einer der beiden Nachbarn von w auf C
  ersetze in C die Kante $\{u,w\}$ durch die Kantenfolge $\langle \{u,v\}, \{v,w\} \rangle$
return C

a) Zeigen Sie, dass der Algorithmus von Prim (beginnend mit Startknoten s) in jedem Schritt denselben Knoten zum aktuellen Baum hinzufügt wie Algorithmus CompleteHamilton zum Kreis C. **2 Punkte**

   Hinweis: Sie dürfen vernachlässigen, dass es in einem Schritt mehrere gleich gute Wahlmöglichkeiten geben kann.

b) Zeigen Sie, dass der Algorithmus CompleteHamilton Güte 2 besitzt. **4 Punkte**

   Hinweis: Verwenden Sie Teilaufgabe a) sowie die *Dreiecksungleichung*.

Aufgabe 4 – Längste Wege

Das Problem LÄNGSTER s-t-WEG besteht darin in einem gegebenen ungerichteten Graphen zu einem gegebenen Paar $\{s,t\}$ von Knoten mit $s \neq t$ einen längsten einfachen Weg zu finden. Zur Erinnerung: ein *Weg der Länge* $k \geq 0$ ist eine Folge $\langle v_0, \ldots, v_k \rangle$ mit der Eigenschaft, dass $\{v_0,v_1\}, \{v_1,v_2\}, \ldots, \{v_{k-1},v_k\}$ Kanten des Graphen sind. Ein Weg ist *einfach*, wenn er jeden Knoten höchstens einmal durchläuft.

Wir wollen zeigen, dass LÄNGSTER s-t-WEG NP-schwer ist, indem wir das Problem HAMILTONWEG auf LÄNGSTER s-t-WEG reduzieren.

a) Sei $G = (V,E)$ ein ungerichteter Graph. Wir erweitern G zu einem Graphen $G' = (V',E')$, dessen Knotenmenge $V' = V \cup \{s,t\}$ um zwei zusätzliche Knoten s und t ergänzt wurde. Außerdem ist $E' = E \cup \{\{s,v\} \mid v \in V\} \cup \{\{v,t\} \mid v \in V\}$, d. h. s und t sind jeweils zu jedem Knoten von G adjazent.

   Erklären Sie, wie man aus einem einfachen Weg der Länge k in G einen einfachen s-t-Weg der Länge $k+2$ in G' erhält. Erklären Sie ebenfalls, wie man aus einem

einfachen s-t-Weg der Länge k in G' einen einfachen Weg der Länge $k-2$ in G konstruieren kann (zwischen beliebigen Knoten aus V). **2 Zusatzpunkte**

b) Zeigen Sie: Es gibt einen Hamiltonweg (d. h. einen einfachen Weg der Länge $n-1$) in G genau dann, wenn es einen einfachen s-t-Weg der Länge $n+1$ in G' gibt. **1 Zusatzpunkt**

c) Zeigen Sie damit: LÄNGSTER s-t-WEG ist NP-schwer. **2 Zusatzpunkte**

Hinweis: Das wichtigste haben wir in der vorherigen Teilaufgabe schon bewiesen. Es fehlen nur noch ein paar Details für eine zulässige Polynomialzeitreduktion.

Bitte geben Sie Ihre Lösungen bis **Dienstag, 12. Mai 2026, 13:00 Uhr** einmal pro Gruppe über WueCampus als PDF-Datei ab. Geben Sie stets die Namen aller an, die das Übungsblatt bearbeitet haben (max. 2).

Begründen Sie Ihre Behauptungen und kommentieren Sie Ihren Pseudocode!

Plagiate werden mit 0 Punkten für das ganze Übungsblatt gewertet.