Zweite Runde Theorie

Aufgaben

Swiss Olympiad in Informatics

3. März 2018

Anweisungen

- Öffne die Prüfungs erst, wenn du dazu aufgefordert wirst. Die Prüfung beginnt für alle Teilnehmer gleichzeitig und dauert 5 Stunden.
- Mit Ausnahme von Uhren (keine Smartwatches) sind keine elektronischen Geräte auf den Tischen erlaubt. Schalte dein Mobiltelefon aus.
- Beginne jede Aufgabe auf einer neuen Seite und schreibe deinen Namen auf alle Blätter. Nummeriere deine Blätter und sortiere sie vor der Abgabe.
- Verwende keinen Bleistift und schreibe nicht mit rot.
- Schreibe lesbar.
- Schau dir jede Teilaufgabe an, auch wenn du eine vorherige nicht lösen konntest. Einige Teilaufgaben lassen sich auch ohne vorherige Teilaufgabe lösen.

Bewertung

Deine Lösung wird anhand ihrer Korrektheit und der asymptotische Laufzeit und Speichernutzung bewertet, sowie der Begründung dieser Punkte. Wir erwarten, dass du einen Beweis oder eine Beweisskizze für die Korrekteit und Laufzeit/Speichernutzung lieferst.

Falls ein Algorithmus verlangt ist, solltest du dich als Richtlinie an folgende Struktur halten:

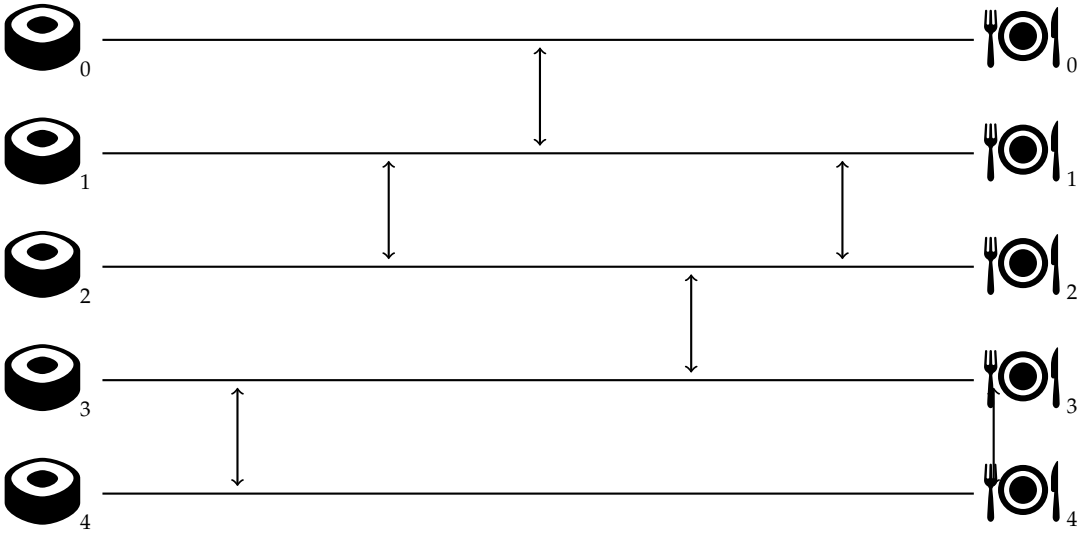
1. Beschreibe die Idee hinter deinem Algorithmus so verständlich wie möglich.
2. Erkläre, wieso der Ansatz die Aufgabe korrekt löst.
3. Analysiere die asymptotische Laufzeit und den asymptotischen Speicherverbrauch.
4. Schreibe ein Lösungsprogramm in Pseudocode auf. Du kannst einfache Teile wie Eingabe/Ausgabe weglassen und auch mathematische Ausdrücke benutzen.

Wenn ein Teil deiner Lösung für mehrere Teilaufgaben gilt, reicht es, ihn einmal aufzuschreiben und von da an auf ihn zu verweisen.

Viel Erfolg!

Sushi Belts

Im Restaurant Takano sind b Fliessbänder montiert, welche Sushi zu den Gäste liefern. Diese Fliessbänder befördern Sushi von Westen nach Osten auf einer gerade Linie. Sie sind nummeriert von 0 bis $b - 1$ von Norden nach Süden.



Am westlichen Ende eines jeden Fliessbandes steht ein Sushi-Chef. Jeder Chef hat sein eigenes Gericht, für welches er bekannt ist (also serviert jedes Fliessband ein anderes Gericht). Am östlichen Ende jedes Fliessbandes befindet sich ein Sitzplatz für einen Gast. Da die Bänder weit auseinander liegen, kann Gast i nur Gerichte vom Band i nehmen. Dies würde aber schnell langweilig für die Gäste, weil sie ja nur von genau einem Gericht essen könnten.

Deshalb hat das Restaurant w Mitarbeiter angestellt, die sich jeweils an ihrer x-Koordinate x_j zwischen Bändern y_j und y_{j+1} befinden. Jeder Mitarbeiter kann von seinen Bändern y_j und y_{j+1} Gerichte vertauschen. Du kannst annehmen, dass sich die Mitarbeiter an unterschiedlichen x-Koordinaten befinden, es kann aber sein, dass mehrere Mitarbeiter zwischen zwei Bänder stehen.

Falls ein Gast ein Gericht bestellt, versuchen alle Mitarbeiter, sich so zu koordinieren, dass sie das Gericht dem Gast liefern können (falls dies möglich ist), indem sie das Gericht von Fliessband zu Fliessband bewegen.

Teilaufgabe 1: Serviere alle Mahlzeiten an alle Gäste (10 Punkte)

Finde für $b = 10$ Sushi Fliessbänder eine Platzierung von 47 oder weniger Mitarbeitern, sodass jede Mahlzeit an jeden Gast gesendet werden kann.

Teilaufgabe 2: Minimale Anzahl Angestellte (30 Punkte)

Takano würde gerne so wenig Mitarbeiter wie möglich anstellen. Gleichzeitig will sie, dass es möglich bleibt, allen Gästen jedes Gericht zu servieren. Für ein gegebenes $b \geq 2$, was ist die minimale Anzahl Mitarbeiter (als eine Funktion von b) die Takano anstellen muss und wie müssen sie platziert werden?

Für den Beweis, dass deine Lösung wirklich der minimalen Anzahl Angestellten entspricht, erhältst du 20 Punkte.

Teilaufgabe 3: Suboptimale Angestellte (60 Punkte)

Die Angestellten in Takano waren nicht in der Lage die vorherigen Aufgaben zu lösen. Also stehen sie einfach an ihrer Position herum und versuchen die Gerichte so gut wie es geht zu liefern.

Gegeben sind b, w, sowie die Zahlen x_j, y_j, welche die Platzierung der Mitarbeiter darstellen. Sei m_i die Anzahl unterschiedlicher Gerichte, die der Gast auf Fliessband i erreichen kann. Beschreibe nun einen Algorithmus der alle $m_0, \ldots, m_{b-1}$ berechnet.

Für den Korrektheitsbeweis deiner Lösung erhältst du 20 Punkte.

Osaka Tube

Maus Stofl ist gerade in Osaka angekommen und steht vor dem Kauf einer SOI-Card (Smart Osaka Inter Card) für den hiesigen ÖV. Die Stadt besteht aus n U-Bahn Stationen, die durch m Linien verbunden sind. Eine Linie wird ohne Halt jeweils zwischen zwei Stationen betrieben und verkehrt in beiden Richtungen. Für jede Linie ist die Fahrtdauer angegeben, die unabhängig von der Richtung ist. Wegen der perfekten Ausführung der Stationen kann die für einen Umstieg benötigte Zeit vernachlässigt werden. Es ist auch möglich, von jeder Station zu jeder anderen Station zu gelangen (möglicherweise durch mehrere Zwischenstopps).

Der Preis für eine Fahrt wird wie folgt bestimmt: jede Station ist genau einer der insgesamt z Zonen zugeteilt. Die Fahrt in der Zone i kostet p_i Yen. Für eine Route zwischen zwei Stationen, die durch mehrere Zwischenstopps führt, wird der Preis aller Zonen auf der Route aufsummiert (dabei wird jede Zone nur einmal belastet). Beispielsweise bei einer Fahrt durch Zonen 0, 0, 3, 0, 4, 4, 3 beträgt der Gesamtpreis $p_0 + p_3 + p_4$.

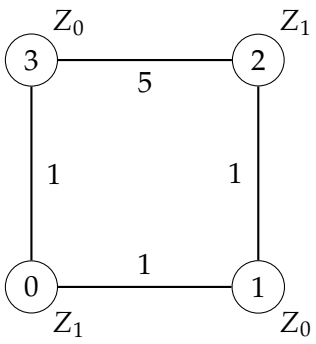
Die *billigste* Route ist eine Route mit dem günstigsten Preis.

Wer mit der SOI-Card unterwegs ist, kann von dem folgenden Angebot profitieren: man steigt an einer Station ein, reist dann beliebig innerhalb dem ÖV-Netz und steigt an einer Station aus. Der Betrag, der der SOI-Card belastet wird, ergibt sich dann als der Preis einer billigsten Route zwischen dem Einstieg- und Ausstiegsort.

Maus Stofl gefällt das Angebot, fragt sich jedoch, ob es konsistent ist. Das Netz ist *konsistent*, wenn es keine zwei Stationen gibt, zwischen denen jede kürzeste Route (bzgl. Fahrtdauer) strikt teurer ist als eine billigste Route zwischen den beiden Stationen.

Da das ÖV-Netz von Osaka ziemlich kompliziert ist, bittet dich Maus Stofl um deine Hilfe.

Ein Beispiel eines Netzes ist auf dem nächsten Bild dargestellt. Es gibt zwei Zonen ($z = 2$), die jeweils ausserhalb des Knotens vermerkt sind. Die Fahrtdauern sind an den Kanten markiert. Die Zonen kosten $p_0 = 1$ Yen und $p_1 = 2$ Yen. Die einzige kürzeste Route von 3 nach 2 ist die Verbindung $3 \to 0 \to 1 \to 2$. Eine billigste Route zwischen 3 und 2 ist die direkte Verbindung $3 \to 2$ oder auch die Verbindung $3 \to 0 \to 1 \to 2$.



Teilaufgabe 1: Ein Gegenbeispiel finden (10 Punkte)

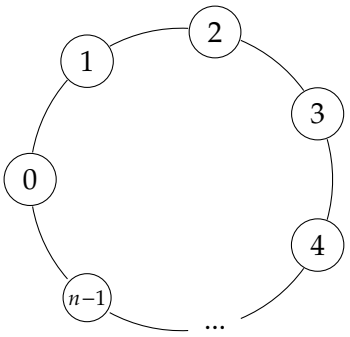
Finde ein mögliches ÖV-Netz und ein zugehöriges Zonensystem, das *nicht* konsistent ist. Das ÖV-Netz soll als ein ungerichteter Graph angegeben werden. Direkt im Graph soll die Fahrtdauer an jeder Kante vermerkt werden (beachte, dass die Fahrtdauer von der Richtung unabhängig ist). Ausserdem soll an jedem Knoten eine Zone vermerkt werden. Schliesslich ist in einer Tabelle der Preis jeder Zone anzugeben. Da jede Zone nur einmal gelöst werden muss, notiere die Zonen an den Knoten (und nicht die Preise), wie in der obigen Abbildung.

Teilaufgabe 2: Ringnetz (30 Punkte)

Nehmen wir für den Moment an, dass das ÖV-Netz ein Kreis ist. D.h. die Station i ist mit der Station $i + 1$ verbunden, für $0 \leq i < n - 1$, und die Station $n - 1$ ist mit der Station 0 verbunden.

Entwerfe einen Algorithmus mit einer möglichst guten Laufzeit und Speichernutzung, der entscheidet, ob ein solches gegebenes Netz konsistent ist.

Ein solches Ringnetz ist auf dem folgenden Bild dargestellt.



Teilaufgabe 3: ÖV-Netz von Osaka (60 Punkte)

Schauen wir uns jetzt das eigentliche ÖV-Netz von Osaka an. Es ist nicht unbedingt ein Kreis, besteht aber nicht aus vielen Zonen (im Vergleich zu n).

Entwerfe einen Algorithmus, der es entscheidet, ob ein solches gegebenes ÖV-Netz konsistent ist. Optimiere Laufzeit und Speichernutzung in erster Linie in Abhängigkeit von n und m, und erst in zweiter Linie auf z.

Dandy Show

Maus Stofl ist wahrlich berühmt in seinem Universum und trägt zu recht den Titel Space Dandy. In einem seiner unzähligen Abenteuer hat er herausgefunden, dass sein Universum einem 2-dimensionalen Polygon entspricht (er kann zumindest nicht diese linienförmigen Wände durchdringen, die er auf seiner Karte aufgezeichnet hat).

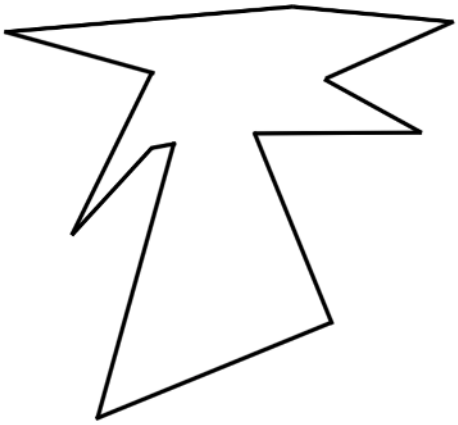
Er ist an Ruhm und Reichtum gekommen und möchte nun irgendwie nicht in Vergessenheit geraten. Dafür hat er auch schon den perfekten Plan: Er möchte die grösste Discoparty seines Universums schmeissen. Damit die Party auch wirklich im Gedächtnis aller Teilnehmer bleibt, möchte er diese an einer Stelle im Universum machen, wo jeder ihn sehen kann! Hilf Stofl, eine solche Stelle in seinem Universum zu finden.

Formale Beschreibung Gegeben ein 2-dimensionales einfaches Polygon (einfach bedeutet, dass die Kanten des Polygons sich nicht schneiden und nur in den Eckpunkten jeweils berühren), finde einen Punkt in diesem Polygon, so dass man mit einer geraden Linie jeden Punkt im Polygon mit einer geraden Linie verbinden kann, ohne die Wand des Polygons zu berühren/schneiden.

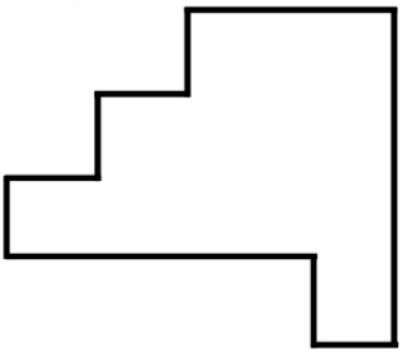
Teilaufgabe 1: Lösen anhand zweier konkreten Universen (10 Punkte)

Markiere alle Stellen, wo Dandy seine Discoparty veranstalten kann.

1.



2.



Teilaufgabe 2: Lösen von achsenparallelen Universen (35 Punkte)

Konstruiere einen Algorithmus, der entweder einen Punkt im Universum findet, in dem Dandy seine Discoparty organisieren kann, oder "too bad" ausgibt, wenn kein solcher Punkt existiert. Das Polygon ist gegeben durch eine Liste von Punkten, die die Eckpunkte des Polygon dem Uhrzeiger nach beschreibt. Dabei hat er herausgefunden, dass alle Kanten des Polygonuniversums entweder horizontal oder vertikal sind (das hat er herausgefunden, indem er nur 90° und 270° Winkel an den Polygonecken gemessen hat. Wir nehmen zusätzlich an, dass der Raum nicht gekrümmt ist).

Teilaufgabe 3: Lösen vom allgemeinen Fall (55 Punkte)

Löse nun den allgemeinen Fall, wo das Universum jedes einfache Polygon sein kann.

Erdgasversorgung

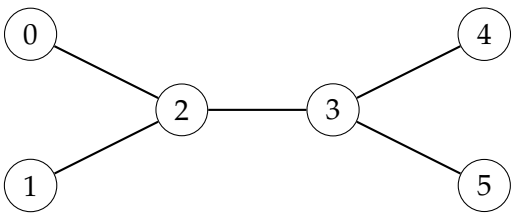
Maus Stofl hat einen Karrieresprung gemacht und ist nun für die Erdgasversorgung in Tsukuba verantwortlich. Das Erdgasnetz besteht aus Knotenpunkten und Leitungen, die diese Knotenpunkte verbinden. Das Netzwerk ist verbunden und es enthält keine Zyklen. Zustätzlich gilt, dass in einem Knotenpunkt maximal 3 Leitungen zusammenkommen. In der Sprache der Graphentheorie heisst dies, dass das Netzwerk ein Baum mit Maximalgrad 3 ist.

Leider hat Maus Stofl schon an seinem ersten Arbeitstag ein Problem. Der Gasbedarf steigt sprunghaft an, was nur bedeuten kann, dass es an einem Knotenpunkt ein Leck hat, über das eine grosse Menge Gas entweicht. Er weiss allerdings nicht, um welchen Knotenpunkt es sich handelt. Um das herauszufinden, entwickelt er einen Plan. In der Nacht wird kein Erdgas verbraucht. Wegen des Überdrucks und da nun irgendwo ein Leck ist, fliesst von jedem Knoten Erdgas in Richtung des Lecks. Maus Stofl kann an einer Leitung ablesen, in welche Richtung das Gas darin fliesst. Das Ablesen an einer Leitung ist aber sehr mühsam und zeitaufwändig. Er möchte nun wissen, wie viele Leitungen er maximal ablesen muss, bis er das Leck gefunden hat.

Hinweis: Du sollst für den Fall optimieren, bei der die Anzahl Knotenpunkte n sehr gross ist. Es ist also zum Beispiel besser, $\sqrt{n} + 100$ als $2 \cdot \sqrt{n} + 2$ Abfragen zu haben. Terme abgesehen vom dominanten Term werden bei der Bewertung ignoriert, also wären $n^{\frac{1}{2}} + 6n^{\frac{1}{4}} + 9$ gleich gut wie $\sqrt{n} - 42$. Wenn du nicht sicher bist, schreibe einfach die gesamte Formel auf.

Formale Beschreibung Gegeben ist ein Baum T mit Maximalgrad 3. An genau einem Knoten befindet sich ein Leck. Wenn es eine Leitung zwischen den Knotenpunkten A und B gibt, dann kanst du die Anfrage (A, B) stellen. Die Antwort ist entweder A, falls das Gas von B nach A fliesst, oder B, falls es in die andere Richtung fliesst. Minimiere die maximale Anzahl an Anfragen die gestellt werden müssen bis das Leck gefunden wurde.

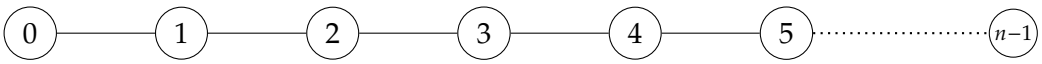
Teilaufgabe 1: Lösen eines Beispiels (10 Punkte)



Mit wie viele Anfragen kann das Leck im abgebildeten Erdgasnetz in jedem Fall gefunden werden? Erkläre kurz, welche Anfragen gestellt werden müssen um diese Anzahl Anfragen nicht zu überschreiten.

Teilaufgabe 2: Ein einfaches Netzwerk (30 Punkte)

Betrachten wir zuerst den einfachern Fall, wenn das Netwerk nur Knotenpunkte hat in denen entweder ein oder zwei Leitungen enden. In anderen Worten, der Maximalgrad des Baumes ist 2. Du kannst ausserdem annehmen, dass Knotenpunkt i mit Knotenpunkt $i + 1$ verbunden ist, für $0 \leq i < n - 1$. Das Netzwerk sieht also so aus:



Du sollst die Anzahl Anfragen angeben, die nötig sind, um das Leck zu finden. Beweise, dass es mit dieser Anzahl Anfragen immer möglich ist, das Leck zu finden. Entwickle einen Algorithmus, der die Anfragen für Maus Stofl stellt und bestimme seine Laufzeit.

Teilaufgabe 3: Der allgemeine Fall (60 Punkte)

Bestimme nun für den Fall, dass der Maximalgrad des Baums 3 ist, die Anzahl nötiger Anfragen. Beweise, das es mit dieser Anzahl Anfragen immer möglich ist, das Leck zu finden. Entwickle einen Algorithmus, der die Anfragen für Maus Stofl stellt und bestimme seine Laufzeit.