1 Weitere Algorithmen für das TSP

Wir betrachten nun noch zwei weitere Algorithmen für das Traveling Salesman Problem. Beim ersten handelt es sich wiederum um eine Variante eines Opt–Schrittes, die empirisch getestet recht gute Ergebnisse liefern soll.

Der zweite Algorithmus dagegen ist empirisch etwas schlechter, allerdings erlaubt er eine Abschätzung seines Verhaltens im *Worst Case*. Genauer kann gezeigt werden, dass für das euklidische TSP die resultierenden Lösungen unabhängig von der Eingabeinstanz höchstens zweimal so lang wie die Optimallösung sind. Es wird also garantiert, dass auch bei ungünstigen Problemen niemals eine wirklich schlechte Lösung erzeugt wird. Liegen derartige Schranken für die maximale Abweichung vor, spricht man auch von *Approximationsalgorithmen*. Der Algorithmus von *Christophides* wird dann zu einer noch geringeren maximalen Abweichung vom Optimum führen.

1.1 Or-opt

Das Verfahren *Or-Opt* ist eine vereinfachte Variante des 3–Opt–Schrittes. Bei diesem Verfahren wird eine Folge von $s = 1$, 2 oder 3 aufeinanderfolgenden Städten zwischen zwei Städte eingesetzt. Dabei wird erst $s = 3$, dann $s = 2$ und schließlich $s = 1$ gesetzt. Für $s = 3$ wird jeder String von 3 in der Tour aufeinanderfolgenden Städten l, m, n zwischen jedes Paar i, j von aufeinenderfolgenden Städten eingesetzt. Der Einsetzvorgang wird durchgeführt, wenn die Gesamtkosten der zu entfernenden Kanten $\{a, l\}$, $\{n, b\}$ und $\{i, j\}$ höher sind, als die Kosten der einzusetzenden Kanten $\{i, l\}$, $\{n, j\}$, $\{a, b\}$. Danach werden alle Folgen von 2 aufeinanderfolgenden Städten betrachtet und dann alle aus einer Stadt bestehenden Folgen. Das Verfahren terminiert, wenn keine Verbesserungen mehr möglich sind. Es kann natürlich auch mit anderen Verfahren, wie Simulated Annealing und genetischen Algorithmen kombiniert werden.

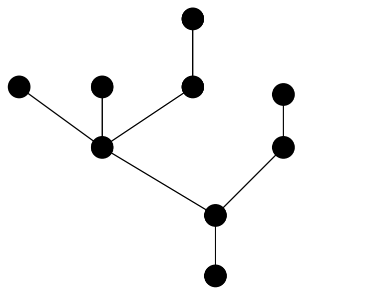
1.2 Minimale Spannbäume und TSP: Ein Approximationsalgorithmus

In diesem Abschnitt betrachten wir einen Algorithmus mit einer Approximationsgüte von 2. Dies bedeutet, dass die vom Algorithmus erzeugte Rundreise für jede Eingabeinstanz I höchstens doppelt so lang werden kann wie die optimale Rundreise. Bezeichnet $Opt(I)$ die Länge einer optimalen Rundreise für ein Eingabeproblem I und $Alg(I)$ die Länge der durch den Algorithmus erzeugten Reise, so gilt also für jede Eingabeinstanz

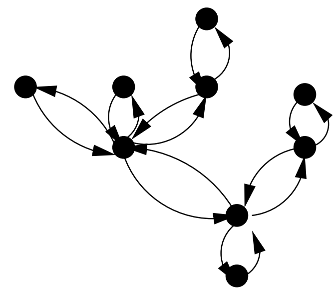
$$\frac{Alg(I)}{Opt(I)} \leq 2$$

. Allerdings machen wir an die Eingabeinstanzen eine Einschränkung: Es muss die *Dreicksungleichung* gelten. Wenn die Kantenlänge zwischen zwei Städten i und j mit c_{ij} bezeichnet wird, so muss für alle $i, j, k \in \{1, \ldots, n\}$ gelten

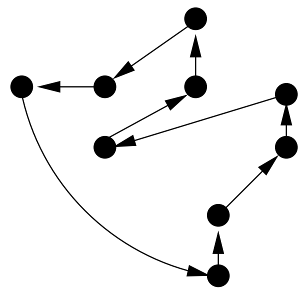
$$c_{ik} \leq c_{ij} + c_{jk}.$$



Minimaler Spannbaum T



Tiefensuche durch T



Tiefensuche durch T mit ‘Shortcuts’

Abbildung 1: Minimaler Spannbaum und induzierte Rundreise

Anders ausgedrückt: Der kürzeste Abstand zwischen zwei Städten ist eine gerade Linie. Die Dreiecksungleichung ist natürlich im euklidischen Fall, bei dem die Kantenkosten c_{ij} genau durch die Distanz zwischen i und j gegeben sind, trivialerweise erfüllt.

Wir versuchen nun, einen Bezug zwischen minimalen Spannbäumen und dem Problem des Handlungsreisenden herzustellen. Ein Spannbaum für eine Menge von n Städten ist eine Auswahl von $n - 1$ Kanten, die alle Städte einer Zusammenhangskomponente miteinander verbindet. Beim Entwickeln von Heuristiken sind wir nun oft mit der Frage konfrontiert, wie gut eine Rundreise sein kann. Wie wir bereits wissen, können wir diese Frage in der Regel nicht beantworten, jedoch können wir mittels des minimalen Spannenbaumes eine untere Schranke für die Länge der optimalen Rundreise bestimmen.

Man sieht nämlich sofort, dass wenn eine Kante aus einer optimalen Rundreise entfernt wird, eine Spannbaum resultiert. Er besteht aus einem einzelnen Pfad, der alle Städte verbindet. Der minimale Spannbaum kann also nur noch kürzer sein. D.h, das Kantengewicht eines minimalen Spannbaumes liefert direkt eine untere Schranke für die optimale Lösung. Darüber hinaus gelingt es uns aber auch mittels des minimalen Spannbaumes eine obere Schranke für die Optimallösung zu bestimmen.

Wir nehmen einmal an, dass wir alle Städte besuchen wollen, allerdings dürfen wir nur

Kanten des minimalen Spannbaums benutzen (vgl. Abb. 1). Eine Möglichkeit wäre, bei der Wurzel des Baumes zu starten und die folgende Strategie zu verfolgen: Falls es eine noch nicht besuchte Kante von dem aktuellen Knoten aus gibt, so folge dieser Kante zum neuen Knoten. Wenn alle Kanten vom aktuellen Knoten aus bereits besucht worden sind, gehe die Kante zurück, über die der aktuelle Knoten als erstes erreicht wurde. Die Prozedur wird beendet, wenn der Ausgangsknoten wieder erreicht wurde. Sie entspricht einer Tiefensuche durch den minimalen Spannbaum.

Offensichtlich wird jeder Knoten einmal besucht und jede Kante wird maximal zweimal durchlaufen. Dadurch ensteht also eine Rundreise durch alle Städte und die Länge der Rundreise ist nicht länger als zwei mal das Gewicht des minimalen Spannbaumes. Da das Gewicht eines minimalen Spannbaumes aber gerade eine untere Schranke für die Länge der optimalen Rundreise ist, haben wir also eine Rundreise erzeugt, die höchstens zweimal so lang wie die optimale Rundreise ist. Wir haben allerdings noch ein Problem: Verschiedene Städte werden möglicherweise mehr als einmal durchlaufen.

Hier kommt nun die Dreiecksungleichung ins Spiel. Wir können nämlich Wiederholungen von Städten vermeiden, wenn wir ‘Abkürzungen’ nehmen. Wir starten dazu genauso wie bisher. Wenn wir allerdings bei der Tiefensuche auf eine bereits betrachtete Stadt stoßen, dann lassen wir diese aus und gehen direkt zur nächsten noch nicht betrachteten Stadt. Aufgrund der Dreiecksungleichung kann dieser direkte Weg nicht länger werden, als der bisherige Weg. Wenn alle Städte besucht sind, dann gehen wir zum Ausgangspunkt zurück. Insgesamt erzeugen wir so eine Rundreise, deren Länge höchstens zweimal der Länge einer optimalen Rundreise beträgt.

Es ist auch interessant sich noch einmal über den Schwierigkeitsgrad der Probleme Gedanken zu machen. Wenn man aus einer Rundreise eine Kante herausnimmt entsteht ein Spannbaum. Minimaler Spannbaum und TSP sind also in gewisser Weise ähnlich. Jedoch kann das Problem der Berechnung eines optimalen (also minimalen) Spannbaumes mit einem einfachen Greedy–Algorithmus gelöst werden, wohingegen die Berechnung einer optimalen Rundreise im schlimmsten Fall gar nicht gelingt. Man sieht den Problemen ihren Schwierigkeitsgrad also ohne weiteres gar nicht an.

1.3 Das Verfahren von Christophides

Eine Eulertour in einem Graphen ist eine Rundreise, die jede Kante des Graphen genau einmal durchläuft. Knoten dürfen mehrmals durchlaufen werden. Wie beispielsweise aus der Vorlesung *Diskrete Strukturen* bekannt, gibt es genau dann eine Eulertour, wenn jeder Knoten einen geraden Grad hat.

Wenn wir uns an das Verfahren erinnern, das einen minimalen Spannbaum T zugrundelegt, so werden mehrfach vorkommende Knoten durch ‘Abkürzungen’ eliminiert. Auf die gleiche Weise kann aus einer Eulertour auch eine zulässige Rundreise für das TSP gemacht werden. Wenn bei einem minimalen Spannbaum wie oben jede Kante einmal verdoppelt wird, liegt sogar ein Eulergraph vor. Die zentrale Idee, um zu einer verbesserten Approximationsgüte zu gelangen, ist, einen besseren Eulergraphen zu finden. Diesen erhalten wir über ein Matching *minimalen* Gewichts.

Sei dazu eine *gerade* Anzahl von Knoten gegeben, die alle durch Kanten miteinander verbunden sind (es liegt also ein vollständiger Graph vor). Dann ist ein Matching minimalen Gewichts M ein Matching auf den Knoten, so dass jeder Knoten zu einer Matchingkante gehört und das Gesamtgewicht der Matchingkanten minimal ist. Es gibt also kein weiteres Matching auf den Knoten mit einem geringeren Kantengewicht. Ein solches Matching kann in $\mathcal{O}(n^3)$ bestimmt werden.

Nun gehen wir wieder zum minimalen Spannbaum zurück, aus dem wir einen Eulergraphen erzeugen wollen. Nach obigen Überlegungen sind nur die Knoten ungeraden Grades zu verändern. Da die Summe aller Knotengrade bekanntlich gerade ist, kann es des Weiteren nur eine gerade Anzahl an Knoten mit ungeradem Grad geben. Man erhält also einen Eulergraphen, wenn man ein Matching auf den Knoten ungeraden Kanten hinzufügt (vrgl. Abb. **??**), da sich der Knotengrad jedes gematchten Knotens um eins erhöht.

In dem so entstandenen Eulergraph kann nun leicht eine Eulertour gefunden werden, in der einige Knoten mehrmals besucht werden können. Aus einer solchen Tour kann leicht eine Rundreise erzeugt werden, indem Knoten die Knoten aus der Eulertour, die nicht zum ersten Mal besucht werden, einfach übersprungen werden. Aus der Dreiecksungleichung folgt, dass die Rundreise dadurch nicht länger wird als die Eulertour.

Insgesamt erhalten wir den folgenden Algorithmus von Christophides:

(1) Bestimme in $O(n^2)$ einen minimalen Spannbaum T.

(2) Erzeuge in $O(n^3)$ ein Minimum Matching M auf der Menge aller Knoten in T mit ungeradem Grad.

(3) Finde eine Eulertour für den Eulergraphen, der als Vereinigung von T und M entsteht. Erzeuge daraus eine zulässige Rundreise durch das Verwenden von Abkürzungen (Shortcuts).

Wir werden nun zeigen, dass die Länge der mit dem Algorithmus von Christophides berechneten Rundreise höchstens $3/2$ mal so lang ist wie die Länge der kürzesten Rundreise:

Sei $c(T)$ die Summe der Kantenlängen des Baumes und sei $c(M)$ die Summe der Kantenlängen des Matchings. Es ist nun klar, dass die für Länge $Alg(I)$ der mit Hilfe des Algorithmus berechneten Tour

$$Alg(I) \leq c(T) + c(M)$$

gilt. Weil T ein minimaler Spannbaum ist, wissen wir außerdem, dass $c(T) \leq OPT(I)$. Wenn wir nun noch zeigen, dass $c(M) \leq 1/2OPT(I)K$ haben wir bewiesen, dass der Algorithmus von Christophides eine $3/2$-Approximation liefert. Dazu betrachten wir die Knoten, die in T einen ungeraden Grad haben in der Reihenfolge, in der sie in einer optimalen Rundreise auftreten; sei $(v_{i_1}, \ldots, v_{i_k})$ diese Reihenfolge. Da k gerade sein muss, können wir auf den Knoten die folgenden beiden Matchings konstruieren:

$$M_1 := \{(v_{i_1}, v_{i_2}), (v_{i_3}, v_4), \ldots, (v_{i_{k-1}}, v_{i_k})\},$$

und

$$M_2 := \{(v_{i_2}, v_{i_3}), (v_{i_4}, v_5), \ldots, (v_{i_k}, v_{i_1})\}.$$

Mit Hilfe der Dreiecksungleichung sieht man leicht, dass $OPT(I) \geq c(M_1) + c(M_2)$, wobei $c(M_j)$ die Summe der Kantenlängen des Matchings M_j ist ($j = 1, 2$). Da M ein Matching mit minimalem Gewicht ist, gilt somit $OPT(I) \geq 2c(M)$, und wir erhalten insgesamt

$$Alg(I) \leq c(T) + c(M) \leq \frac{3}{2} OPT(I).$$

Zum Schluss sei noch bemerkt, dass es Beispiele gibt, bei denen das Verfahren von Christophides keine bessere Approximation als $3/2$ liefert, die Grenze ist also scharf.