NP-Vollständigkeit
Im Laufe Ihres Informatiker-Lebens wird Ihnen häufig die Aufgabe gestellt, Entscheidungs- bzw. Planungsprobleme zu automatisieren, die bisher ein Mensch manuell durchgeführt hat. Ideal wäre es hierbei, wenn Sie dabei die menschliche Intuition bzw. Planungserfahrung durch eine von ihrem Programm berechnete optimale Entscheidung bzw. Plan ersetzen. Manchmal ist das effizient (in polynomialer Zeit) möglich wie z.B. beim Navigationsproblem durch den Dijkstra-Algorithmus (Dynamisches Programmieren). Meistens ist das Planungsproblem jedoch NP-Vollständig. Dies bedeutet, dass
(i) kein effizienter Algorithmus gefunden werden kann ohne damit bereits das berühmte P=NP-Problem zu lösen,
(ii) bisher nur Algorithmen mit exponentieller Rechenzeit zur Berechnung der optimalen Lösung bekannt sind,
(ii) für die praktische Software-Implementierung deshalb jeweils nur Näherungslösungen gesucht sind: **Finde in vorgegebener Zeit eine möglichst gute Lösung**.

Da der Mensch solch schwierige Entscheidungsproblem genauso wenig in kurzer Zeit optimal lösen kann, ist das Ziel für die Entwicklung von Lösungsverfahren: In wesentlich kürzerer Zeit wesentlich bessere Lösungen als der menschliche Planer zu finden.

Die folgende Aufgabe hilft Ihnen zu entscheiden, ob die Suche nach einem Algorithmus für die optimale Lösung aussichtslos ist, da die NP-Vollständigkeit (leicht) nachgewiesen werden kann.

Aufgabe 1 NP-vollständigkeit von TSP – Varianten in der Anwendung

Zeigen Sie, dass eines der folgenden praxisnahen Auslieferprobleme NP-Vollständig ist. Gehen Sie in folgenden Schritten vor (gemäß den beiden Folien zur formalen Definition von NP und NPC):

(i) Definieren Sie das zugehörige Entscheidungsproblem A
(Für Optimierungsproblem O mit " ... minimiere f(z)"
lautet das Entscheidungsproblem A " ... gibt es Lösung z mit f(z) <= K ")
Hinweis:

- Optimierungsproblem entspricht Funktion: Ausgabe der optimalen Lösung z zu Problem x
- Entscheidungsproblem entspricht Prädikat: 1 ⇔ Es gibt Lösung z zu x mit $f(z) \leq K$

(ii) Problem A liegt in NP: Hierzu müssen Sie kurz ein Prüfprogramm P skizzieren, das zu einer Probleminstanz x eine Lösung z rät und dann prüft ob f(z) unter der Schranke K liegt. Überprüfen Sie, ob die Kodierungslänge von z durch ein Polynom p über die Kodierungslänge der vorgegebenen Probleminstanz x beschränkt ist. Meist ist die Kodierungslänge von einer Lösung z kleiner als die Problembeschreibung x und es ist deshalb p aus O(n).

(iii) Zeigen Sie, dass Ihr Problem NP-hart ist. Es genügt hierzu zu zeigen, dass sich ein geeignetes als NP-vollständig bekanntes Problem B auf ihr Problem A reduzieren lässt. Dieser Nachweis ist sehr einfach, wenn P ein Spezialfall von X ist. Dann genügt als Abbildung f für die Reduktion bereits (fast) die Identität (eventuell unterschiedliche Kodierung beider Probleme beachten!).
In unseren Beispielen ist das TSP-Problem geeignet.

Sie können sich eines der folgenden Auslieferungsprobleme aussuchen oder ähnliche weitere selber definieren ;-):

(i) **TSPmitLKW**
Gegeben: Entfernungsmatrix E, Standorte der n Kunden $k_1, \dots, k_n$, m LKWs $L_1,.., L_m$.
Gesucht: m Rundtouren mit minimaler Gesamtlänge, so dass alle n Kunden beliefert werden.

(ii) **TSPmitLKW+Fahrzeit**
Gegeben: Entfernungsmatrix E, Standorte der n Kunden $k_1, \dots, k_n$, m LKWs $L_1,.., L_m$.
Gesucht: m Rundtouren mit minimaler Gesamtlänge, so dass alle n Kunden beliefert werden und die Fahrtzeit von 8 h nicht überschreitet.

(iii) **TSPmit LKW+Kapazität**
Gegeben: Entfernungsmatrix E, Standorte der n Kunden $k_1, \dots, k_n$, Auslieferungsgewichte der Pakete der n Kunden $g_1, \dots, g_n$, m LKWs $L_1,.., L_m$ mit Zuladekapazität $K_1, \dots, K_m$.
Gesucht: m Rundtouren (je 1 pro LKW) mit minimaler Gesamtlänge, so dass alle n Kunden beliefert werden und die Zuladungskapazität bei keinem LKW überschritten wird.

(iv) **TSPmitLKW+Fahrzeit+Kapazität**
Gegeben: Entfernungsmatrix E, Standorte der n Kunden $k_1, \dots, k_n$, Auslieferungsgewichte der Pakete der n Kunden $g_1, \dots, g_n$, m LKWs $L_1,.., L_m$ mit Zuladekapazität $K_1, \dots, K_m$.
Gesucht: m Rundtouren (je 1 pro LKW) mit minimaler Gesamtlänge, so dass alle n Kunden beliefert werden und weder die Fahrzeit von 8 h noch die Zuladungskapazität bei keinem LKW überschritten wird.

(v) **Detailed Scheduling**
Gegeben: Produktionsaufträge p1, ... , pn, mit Umrüstzeiten von p_i auf p_j in Rüstmatrix R[i,j],
m Maschinen $M_1,.., M_m$ mit Schichtzeiten $T_1, \dots, T_m$.
Gesucht: Reihenfolgeplan auf den m Maschinen mit minimaler Gesamtumrüstzeit unter Einhaltung der Schichtzeiten.

Lösung für (ii): **$\text{TSP}_{\text{mitLKW+Fahrzeit}}$**

Zum Nachweis: $\text{TSP}_{\text{mitLKW+Fahrzeit}}$ in NP

Optimerungsproblem O = (n , m, E , F)
- Entfernungsmatrix: E [nxn] of integer   E(i,j) = Entfernung von Kunde i zu j
- Fahrtzeitvektor: F [m] of integer   F(i) = Fahrzeit von LKW i)

Gesucht ist die Kürzeste Gesamttour, die alle Restriktionen (Fahrzeiten) einhält

Entscheidungsproblem E = { (n , m, E , F, K) |
Es gibt ein Gesamttour mit Länge ≤ K, die alle Restriktionen (Fahrzeiten) einhält }

x = (n , m, E , F, K)
z = Tourplan für alle m LkWs
= $(r_{1,1}, r_{1,2}, \dots, r_{1,n_1})\ (r_{2,1}, r_{2,2}, \dots, r_{2,n_2}) \dots (r_{m,1}, r_{m,2}, \dots, r_{m,n_m})$
mit $r_{i,j}$ = j-ter Kunde für LKW i

z.B.: Tour(1) = (3, 2, 7, 5) , Tour(2) = (4, 6, 8, 1, 11)

Prüfprogramm P(x,z)
Für jeden LKW i berechne Tourlänge
$T[i] = \sum_{j=1}^{n_i} \mathrm{E}(r_{i,j}, r_{i,j+1})$
Überprüfe ob Fahrzeiten eingehalten sind:
$\forall i$: T[i] ≤ F[i]
Überprüfe ob alle Kunden genau einmal besucht werden:
Sort (Tour[1] ,Tour [2] ,…, Tour [m]) = {1,2,3,..,n}
Überprüfe ob die Obergrenze für die Gesamt-Tour-Länge eingehalten ist:
$\sum_{i=1}^{m} \mathrm{T(i)} \leq K$.

Zum Nachweis: $\text{TSP}_{\text{mitLKW+Fahrzeit}}$ ist NP-hart

Reduktion von TSP auf $\text{TSP}_{\text{mitLKW+Fahrzeit}}$:

x aus Entscheidungsproblem TSP: x= (n,E,K)
d.h. es gibt Rundreise für E mit Länge ≤ K
⇔
f(x) = f(n,E,K) = (n , 1, E, n*max {E}, K) aus $\text{TSP}_{\text{mitLKW+Fahrzeit}}$
d.h. es gibt Rundreise für E mit einem LKW mit Länge ≤ K
unter Einhaltung der Fahrzeit n*max {E}

kurz: x ∈TSP ⇔ f(x) ∈ $\text{TSP}_{\text{mitLKW+Fahrzeit}}$

und f ist polynomial berechenbar
(fast die Identität für Parameter n, 1, E und K
nur Aufwand $O(n^2)$ für die Fahrzeitbeschränkung n*max{E})

Aufgabe 2 TSP Problem – Varianten der Kodierung

Betrachten Sie folgende 3 Mengen:

1. A_1 = { (n,E,k) | es gibt Rundtour mit Länge $\leq$k zu E = Entfernungsmatrix [n,n] of integer}
2. A_2 = { 2 n 2 e_{11} 2 e_{12} 2 .. 2 e_{nn} 2 k | es gibt Rundtour mit Länge $\leq$ Dualzahl k
   zu E = Entfernungsmatrix [e_{11} e_{12} … e_{nn}] of Dualzahl}
3. A_3 = { val (2 n 2 e_{11} 2 e_{12} 2 .. 2 e_{nn} 2 k) | es gibt Rundtour mit Länge $\leq$ Dualzahl k
   zu E = Entfernungsmatrix [e_{11} e_{12} … e_{nn}] of Dualzahl}
   dabei berechnet val den Wert einer Ternärzahl

Beachten Sie hierbei: $A_2 \subset \{0,1,2\}^*$ und $A_3 \subset$ |N

Welche der drei Mengen definiert das TSP Problem??

A) Zeigen Sie, dass die 3 Probleme jeweils aufeinander reduzierbar und deshalb gleich schwer zu lösen sind.

B) Welche der drei folgenden Worte x_i liegt in A_2 ?

   x_1 = 2 10 2 0 2 1 2 01 2 0 2 10,
   x_2 = 2 10 2 0 2 1 2 1 2 0 2 1,
   x_3 = 2 10 2 1 2 1 2 1 2 101.

C) Liegt die Zahl 628_{10} in A_3 ?

Lösung

Alle 3 Mengen sind alternative Definitionen des TSP-Problems

A) Um zu zeigen, dass A_i reduzierbar auf A_j ist, muss man nur eine Funktion f definieren, die die Kodierung von A_i in die Kodierung von A_j umformt bzw. übersetzt.
B) Das TSP zu x_1 hat 2 Knoten, die mit Kantenlänge 1 jeweils verbunden sind. Die Rundtour 1-2-1 hat Länge 2. Damit ist die Beschränkung 2 eingehalten und x_1 ist in A_2 .
   Das TSP zu x_2 hat 2 Knoten, die mit Kantenlänge 1 jeweils verbunden sind. Jede Rundtour hat Länge 2. Damit kann die Beschränkung 1 nicht eingehalten werden und x_2 ist nicht in A_2 .
   Das TSP zu x_3 hat 2 Knoten, aber die Entfernungsmatrix nur 3 Einträge. Damit ist die Kodierung nicht korrekt und deshalb x_2 nicht in A_2 .

C) 628 entspricht der Ternärzahl 2 1 2 0 2 1, d.h. einem TSP mit 1 Knoten, Entfernungsmatrix = [0] und die einzig mögliche Rundtour 1 hat Länge 0.
   Damit ist die Beschränkung 1 eingehalten und die Zahl 628_{10} ist in A_3 .