Lagrange-Relaxierung und Subgradientenverfahren

Wir wollen nun eine Methode vorstellen, mit der man gegebene Relaxierungen verbessern kann. Wir werden die Idee zunächst an der 1-Baum-Relaxierung des symmetrischen TSPs erklären (hierbei ist sie auch von Held & Karp (1970) entwickelt worden), und sie dann in voller Allgemeinheit formulieren. Wir beginnen mit einer simplen Beobachtung.

Satz. *Gegeben sei ein symmetrisches TSP mit Entfernungen $c_{ij}, 1 \le i < j \le n$. Seien λ_i, $i = 1, \ldots, n$, Knotengewichte, $\lambda := \sum_{i=1}^{n} \lambda_i$ und*

$$c'_{ij} := c_{ij} + \lambda_i + \lambda_j, \quad 1 \le i < j \le n.$$

Für das modifizierte TSP mit Entfernungen c'_{ij} gilt:

$$\begin{aligned} c'(T) &= c(T) + 2\sum_{i=1}^{n} \lambda_i = c(T) + \lambda && \text{für alle Touren } T, \\ c'(B) &= c(B) + \sum_{i=1}^{n} d_B(i)\lambda_i && \text{für alle 1-Bäume } B, \end{aligned}$$

wobei $d_B(i)$ den Grad des Knotens i im 1-Baum B bezeichnet. Das heißt, die optimalen Touren bezüglich c bleiben auch optimal bezüglich c', während ein optimaler 1-Baum bezüglich c nicht unbedingt optimal bezüglich c' ist. △

Beweis. Offensichtlich. □

Korollar. *Gegeben sei ein symmetrisches TSP mit Entfernungen c_{ij}, $1 \le i < j \le n$. Dann ist*

$$f^* := \max_{\lambda \in \mathbb{R}^n} \Big(\min_{B \text{ 1-Baum}} \big(c(B) + \sum_{i=1}^{n} (d_B(i) - 2)\lambda_i \big) \Big)$$

eine untere Schranke für die Länge der optimalen Tour. △

Beweis. Jede Tour ist ein 1-Baum, und es ist $d_T(i) = 2$ für jeden Knoten i in jeder Tour T. Also gilt für jede Tour T

$$c(T) + \sum_{i=1}^{n} (d_T(i) - 2)\lambda_i = c(T),$$

d. h. für jedes $\lambda \in \mathbb{R}^n$ ist der Wert des minimalen 1-Baumes höchstens so groß wie die Länge der kürzesten Tour. □

Beispiel. Wir betrachten das 6-Städte TSP, das durch die Entferungen in Tabelle 1 gegeben ist. Wir eliminieren Wuppertal und erhalten den in Abbildung 1 dargestellten minimalen 1-Baum mit Länge 403 km. Wir führen als Knotengewichte

$$\lambda_K = 15\,, \quad \lambda_F = -120\,, \quad \lambda_A = -5\,, \quad \lambda_D = 0\,, \quad \lambda_B = 0$$

	A	B	D	F	K	W
Aachen	–	91	80	259	70	121
Bonn	91	–	77	175	27	84
Düsseldorf	80	77	–	232	47	29
Frankfurt	259	175	232	–	189	236
Köln	70	27	47	189	–	55
Wuppertal	121	84	29	236	55	–

Tabelle 1: Entfernungen (km) zwischen 6 Städten im Rheinland

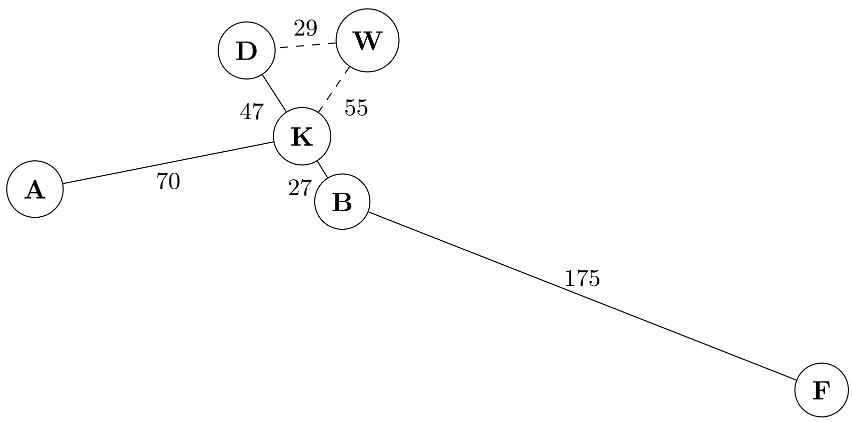

Abbildung 1: Minimaler 1-Baum bezüglich Wuppertal

		-5	0	0	-120	15	0
		A	B	D	F	K	W
-5	A	–	86	75	134	80	116
0	B	86	–	77	55	42	84
0	D	75	77	–	112	62	29
-120	F	134	55	112	–	84	116
15	K	80	42	62	84	–	70
0	W	116	84	29	116	70	–

Tabelle 2: Knotengewichte und modifizierte Entfernungen für 6 Städte im Rheinland

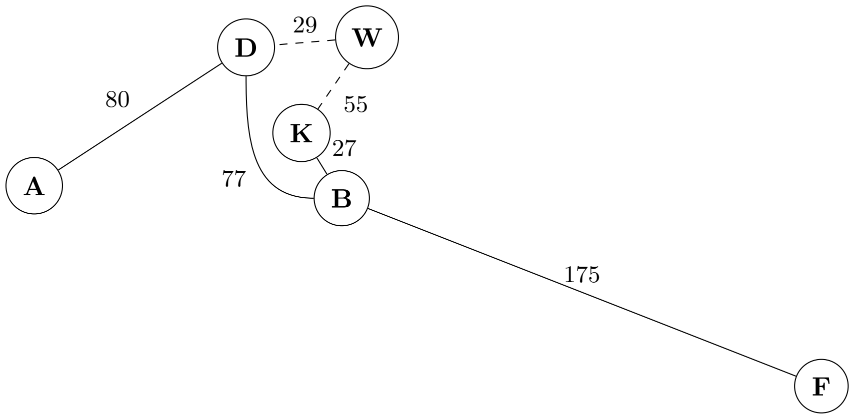

Abbildung 2: Minimaler 1-Baum bezüglich der Knotengewichte in Tabelle 2

ein und erhalten die in Tabelle 2 wiedergegebene Entfernungsmatrix. Der bezüglich der neuen Entferungen kürzeste 1-Baum hat die Länge 443 km und ist in Abbildung 2 dargestellt. Damit haben wir also eine bessere untere Schranke für die optimale Tour gefunden. $\triangle$

Wir wissen zwar, wie wir 1-Bäume schnell berechnen können, aber wie soll das im Korollar formulierte freie Maximierungsproblem (über alle $\lambda \in \mathbb{R}^n$) gelöst werden? Wie finden wir f^*?

Auf folgende Weise können wir eine Funktion $f : \mathbb{R}^n \to \mathbb{R}$ definieren:

$$f(\lambda) := \min_{B \text{ 1-Baum}} \Big(c(B) + \sum_{i=1}^{n} (d_B(i) - 2)\lambda_i \Big) \tag{1.4}$$

d. h. f ist eine Funktion, deren Auswertung die Berechnung der Optimallösung eines Optimierungsproblems erfordert. Das Problem, f^* zu finden, ist damit das unbeschränkte Optimierungsproblem

$$\max_{\lambda \in \mathbb{R}^n} f(\lambda).$$

Um diese Situation zu analysieren, verallgemeinern wir die vorliegende Fragestellung: Wir betrachten ein ($\mathcal{NP}$-schwieriges) ganzzahliges lineares Programm der folgenden Form

$$\begin{aligned} c^* := \min \ & c^T x \\ & Ax = b \\ & Dx \leq d \\ & x \geq 0 \\ & x \text{ ganzzahlig}, \end{aligned} \tag{1.5}$$

wobei A eine (m, n)-Matrix sei. Wir setzen

$$\begin{aligned} P &:= \{x \in \mathbb{R}^n \mid Ax = b, Dx \leq d, x \geq 0, x \text{ ganzzahlig}\}, \\ Q &:= \{x \in \mathbb{R}^n \mid Dx \leq d, x \geq 0, x \text{ ganzzahlig}\}. \end{aligned}$$

Definition. *Die Funktion* $f : \mathbb{R}^m \to \mathbb{R}$ *definiert durch*

$$f(\lambda) := \min\{c^T x + \lambda^T (Ax - b) \mid x \in Q\}$$

heißt Lagrange-Funktion *von* (1.5)*, der Vektor* $\lambda \in \mathbb{R}^m$ *heißt Vektor der* Lagrange-Multiplikatoren. *Das Optimierungsproblem*

$$\max_{\lambda \in \mathbb{R}^m} f(\lambda) \tag{1.7}$$

heißt Lagrange-Relaxierung *von* (1.5). △

Die in (1.4) definierte Funktion ist die Lagrange-Funktion des symmetrischen TSP bezüglich der 1-Baum Relaxierung. Denn, wie wir wissen, ist das folgende Programm eine ganzzahlige Formulierung des TSP:

$$\begin{aligned}
\min \quad & c^T x \\
x(\delta(v)) &= 2 && 1 \leq v \leq n && (1.8)\\
x(E(W)) &\leq |W| - 1 && \forall\, W \subseteq V \setminus \{1\}, |W| \geq 3 && (1.9)\\
x(\{2, \ldots, n\}) &\geq n - 2 && && (1.10)\\
x(\delta(1)) &\leq 2 && && (1.11)\\
x(\delta(1)) &\geq 2 && && (1.12)\\
x &\leq 1 && && (1.13)\\
x &\geq 0 && && (1.14)\\
x & \text{ ganzzahlig.} && && (1.15)
\end{aligned}$$

Schreiben wir das Gleichungssystem (1.8) als $Ax = b$ und die Ungleichungen (1.9)–(1.13) als $Dx \leq d$, so ist die in (1.4) definierte Funktion genau die Lagrange-Funktion des TSP-IPs.

Satz. *Seien Q und P (wie oben definiert) nicht leer und Q endlich. Dann gilt:*

(a) *Die Lagrange-Funktion* $f(\lambda) = \min_{x \in Q}(c^T x + \lambda^T (Ax - b))$ *ist konkav, stückweise linear und nach oben beschränkt.*

(b) *Sei* $\lambda^* \in \mathbb{R}^m$ *mit*

$$f(\lambda^*) = \max_{\lambda \in \mathbb{R}^m} f(\lambda)$$

eine Optimallösung der Lagrange-Relaxierung von (1.5)*, so gilt*

$$f(\lambda^*) \leq c^*,$$

d. h. $f(\lambda^*)$ *ist eine untere Schranke für den Optimalwert von* (1.5). △

Beweis. (a) Wir können für jeden Vektor $x \in Q$ eine affine Funktion g_x in λ wie folgt definieren:

$$g_x(\lambda) := c^T x + \lambda^T (Ax - b).$$

Da Q endlich ist, ist f als Minimum endlich vieler affiner Funktionen darstellbar, d. h.

$$f(\lambda) = \min_{x \in Q} g_x(\lambda).$$

Wir zeigen

- **f ist konkav.**
  Es ist zu beweisen: $f(\alpha\lambda + (1-\alpha)\mu) \geq \alpha f(\lambda) + (1-\alpha) f(\mu)$ für alle $0 \leq \alpha \leq 1$ und für alle $\lambda, \mu \in \mathbb{R}^m$.

$$\begin{aligned} f(\alpha\lambda + (1-\alpha)\mu) &= \min_{x \in Q} g_x(\alpha\lambda + (1-\alpha)\mu) \\ &= \min_{x \in Q} (\alpha g_x(\lambda) + (1-\alpha) g_x(\mu)) \\ &\geq \alpha \min_{x \in Q} g_x(\lambda) + (1-\alpha) \min_{x \in Q} g_x(\mu) \\ &= \alpha f(\lambda) + (1-\alpha) f(\mu). \end{aligned}$$

- **f ist stückweise linear.**
  Sei für jedes $x \in Q$: $L_x := \{\lambda \in \mathbb{R}^m \mid f(\lambda) = g_x(\lambda)\}$. Es gilt natürlich $\bigcup_{x \in Q} L_x = \mathbb{R}^m$. Wenn wir zeigen können, dass L_x konvex ist, dann gilt $f(\lambda) = g_x(\lambda) \; \forall \, \lambda \in L_x$. Somit ist f linear auf L_x, d. h. stückweise linear auf $\mathbb{R}^m$.
  L_x konvex $\iff$ $\alpha\lambda + (1-\alpha)\mu \in L_x$ für alle $0 \leq \alpha \leq 1$ und für alle $\lambda, \mu \in L_x$.

$$\begin{aligned} g_x(\alpha\lambda + (1-\alpha)\mu) \;&\geq\; f(\alpha\lambda + (1-\alpha)\mu) \;\geq\; \alpha f(\lambda) + (1-\alpha) f(\mu) \\ &=\; \alpha g_x(\lambda) + (1-\alpha) g_x(\mu) \;=\; g_x(\alpha\lambda + (1-\alpha)\mu). \end{aligned}$$

  Daraus folgt: L_x ist konvex für alle $x \in Q$, und somit ist f stückweise linear.

- **f ist nach oben beschränkt.**
  Sei $\overline{x} \in P$ beliebig. Dann gilt für jeden beliebigen Vektor von Lagrange-Multiplikatoren $\lambda \in \mathbb{R}^m$:

$$f(\lambda) = \min_{x \in Q} g_x(\lambda) \leq g_{\overline{x}}(\lambda) = c^T \overline{x}.$$

  Daraus folgt insbesondere, dass $f(\lambda) \leq c^*$ für alle $\lambda \in \mathbb{R}^m$ gilt.

(b) ist offensichtlich. □

Es besteht nun das Problem, das Maximum der Lagrange-Funktion f zu bestimmen. Die Standardverfahren der nichtlinearen Optimierung kann man hier nicht anwenden, da die Funktion f i. A. nicht differenzierbar ist. Eine Verallgemeinerung des Gradientenkonzepts aus der Analysis führt jedoch zu einer Lösungsmöglichkeit.

Definition. *Sei $f : \mathbb{R}^m \to \mathbb{R}$ eine konkave Funktion, dann heißt ein Vektor $u \in \mathbb{R}^m$ ein* Subgradient an f im Punkte $\lambda_0 \in \mathbb{R}^m$, *falls gilt*

$$f(\lambda) - f(\lambda_0) \leq u^T(\lambda - \lambda_0) \quad \textit{für alle } \lambda \in \mathbb{R}^m.$$

Die Menge

$$\partial f(\lambda_0) := \{u \in \mathbb{R}^m \mid u \textit{ ist Subgradient an } f \textit{ in } \lambda_0\}$$

heißt Subdifferential *von f in λ_0.* △

Bemerkung. Ist die konkave Funktion f differenzierbar in λ_0, so gilt

$$\partial f(\lambda_0) = \{\bigtriangledown f(\lambda_0)\},$$

wobei $\bigtriangledown f(\lambda_0)$ den Gradienten von f an der Stelle λ_0 bezeichnet. △

Der folgende Satz verallgemeinert die uns aus der Schule bekannte Tatsache, dass ein Punkt eine differenzierbare konkave Funktion f maximiert, wenn die Ableitung von f in diesem Punkt Null ist.

Satz. *Ein Vektor $\lambda^* \in \mathbb{R}^m$ maximiert die konkave Funktion f genau dann, wenn $0 \in \partial f(\lambda^*)$.* △

Beweis. Falls $0 \in \partial f(\lambda^*)$, dann ergibt die Subgradientenungleichung der Definition:

$$f(\lambda) - f(\lambda^*) \leq 0^T(\lambda - \lambda^*) \implies f(\lambda) \leq f(\lambda^*) \text{ für alle } \lambda \in \mathbb{R}^m.$$

Falls $f(\lambda^*)$ maximal ist, so ist $f(\lambda) \leq f(\lambda^*)$ für alle $\lambda \in \mathbb{R}^m$, also $f(\lambda) - f(\lambda^*) \leq 0 = 0^T(\lambda - \lambda^*)$. Daraus folgt $0 \in \partial f(\lambda^*)$. □

Beispiel. Für die konkave, stückweise lineare Funktion

$$f : \mathbb{R}^2 \to \mathbb{R}, (\lambda_1, \lambda_2) \mapsto -(|\lambda_1| + |\lambda_2|)$$

ist das Subdifferential gegeben durch

$$\partial f(\lambda_0) = \begin{cases} [-1,1] \times [-1,1] & \text{falls } \lambda_0 = (0,0)^T, \\ \{(-\operatorname{sign} \alpha, \gamma)^T \mid -1 \leq \gamma \leq 1\} & \text{falls } \lambda_0 = (\alpha, 0)^T, \alpha \in \mathbb{R} \setminus \{0\}, \\ \{(\gamma, -\operatorname{sign} \alpha)^T \mid -1 \leq \gamma \leq 1\} & \text{falls } \lambda_0 = (0, \alpha)^T, \alpha \in \mathbb{R} \setminus \{0\}, \\ \{(-\operatorname{sign} \alpha, -\operatorname{sign} \beta)^T\} & \text{falls } \lambda_0 = (\alpha, \beta)^T, \alpha, \beta \in \mathbb{R} \setminus \{0\}, \end{cases} \quad \triangle$$

siehe Abbildung 3.

Für die speziellen Lagrange-Funktionen, die wir hier betrachten, können wir die Subdifferentiale einfach berechnen.

Satz. *Sei Q nicht leer und endlich und $f(\lambda) := \min_{x \in Q}(c^T x + \lambda^T(Ax - b))$, so gilt folgendes: Setzen wir für $\lambda_0 \in \mathbb{R}^m$, $L_0 := \{x_0 \in \mathbb{R}^n \mid f(\lambda_0) = c^T x_0 + \lambda_0^T(Ax_0 - b)\}$, so ist*

$$\partial f(\lambda_0) = \operatorname{conv}\{Ax_0 - b \mid x_0 \in L_0\}. \quad \triangle$$

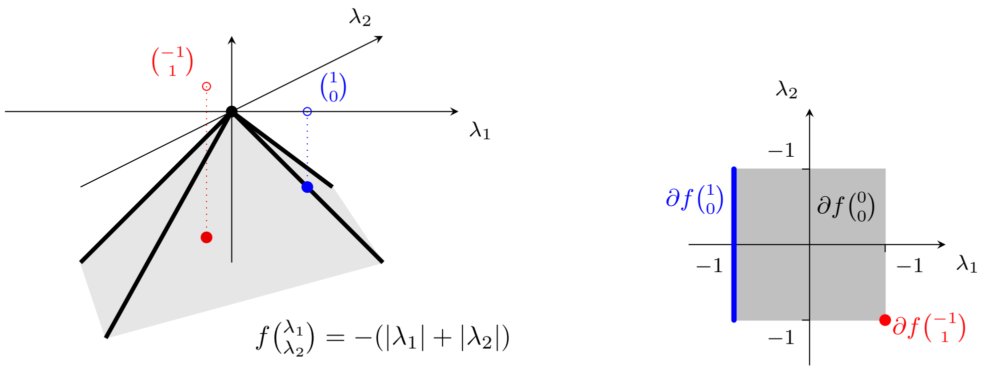

Abbildung 3: Graph einer konkaven Funktion f und einige Subdifferentiale

Aus Zeitgründen können wir auf den Beweis dieses Satzes hier nicht eingehen. Der Satz hat einen einfachen geometrischen Gehalt. Für jedes $\lambda_0 \in \mathbb{R}^m$ ist $\partial f(\lambda_0)$ ein Polytop. Dieses Polytop erhält man wie folgt. Man nehme alle Punkte $x_0 \in \mathbb{R}^n$, die das Minimum des Optimierungsproblems $\min\{c^T x + \lambda_0^T(Ax - b) \mid x \in Q\}$ realisieren, also alle Optimalwerte bezüglich des Vektors der Lagrange-Multiplikation λ_0. Dann bilde man die konvexe Hülle all dieser Punkte und erhält das Subdifferential $\partial f(\lambda_0)$.

Beispiel. Im Falle des symmetrischen TSPs gilt für einen Vektor von Knotengewichten $\lambda := (\lambda_1, \ldots, \lambda_n)$:

$$\partial f(\lambda) = \operatorname{conv}\left\{ \begin{pmatrix} d_B(1) - 2 \\ d_B(2) - 2 \\ \vdots \\ d_B(n) - 2 \end{pmatrix} \mid B \text{ optimaler 1-Baum bezüglich } c' \right\}.$$

Hat man einen optimalen 1-Baum B bezüglich λ, so kann man also einen Subgradienten der Lagrange-Funktion f direkt aus den Knotengraden von B ablesen. $\triangle$

Zur Lösung der Optimierungsaufgabe (1.7) ist die folgende Methode vorgeschlagen worden:

Algorithmus (Allgemeines Subgradientenverfahren).
Eingabe: Konkave Funktion $f : \mathbb{R}^m \to \mathbb{R}$
Ausgabe: $\max\{f(\lambda) \mid \lambda \in \mathbb{R}^m\}$

1. Wähle $\lambda_0 \in \mathbb{R}^m$ beliebig, setze $j := 0$.
2. Berechne $f(\lambda_j)$ und $\partial f(\lambda_j)$.
3. Ist ein (noch zu definierendes) STOP-Kriterium erfüllt?
   Falls ja, so ist eine „gute“ untere Schranke für das Maximum oder das Maximum selbst gefunden. $\longrightarrow$ STOP.
   Falls nein, gehe zu Schritt 4.

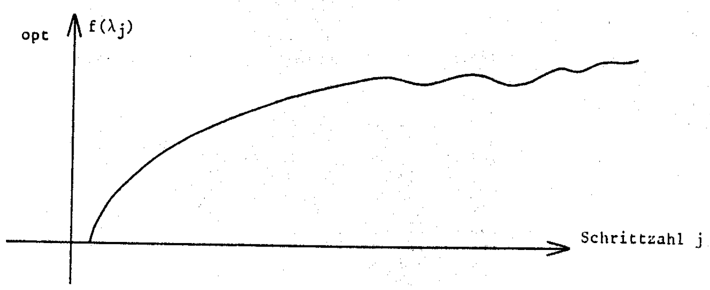

Abbildung 4: Typisches Konvergenzverhalten beim Subgradientenverfahren

4. Wähle neues λ_{j+1}, setze z. B.

$$\lambda_{j+1} := \lambda_j + t_j \, \frac{u_j}{\|u_j\|},$$

wobei $t_j \in \mathbb{R}$ eine Schrittlänge ist und $u_j \in \partial f(\lambda_j)$.

5. Setze $j := j + 1$ und gehe zu Schritt 2. △

Bezüglich dieses Verfahrens kann man Folgendes beweisen.

Satz (Polyak, 1967). *Falls die konkave Funktion $f : \mathbb{R}^m \to \mathbb{R}$ ein Maximum besitzt und falls die Folge $(t_j)_{j\in\mathbb{N}}$ der Schrittlängen die Bedingungen*

(a) $t_j > 0$ *für alle* $j \in \mathbb{N}$,

(b) $\lim_{j\to\infty} t_j = 0$,

(c) $\sum_{j=0}^{\infty} t_j = \infty$

erfüllt, so gilt

$$\lim_{j\to\infty} f(\lambda_j) = \max_{\lambda\in\mathbb{R}^m} f(\lambda).$$

△

Bei konkreten Rechnungen zeigt sich allerdings im allgemeinen, dass die (nach dem Satz von Polyak theoretisch existierende) Konvergenz praktisch nicht erreicht wird. Abbildung 4 zeigt einen typischen Verlauf des Wertes $f(\lambda_j)$, abhängig von der Schrittzahl j.

Mit anderen Schrittlängenparametern kann man praktisch i. A. ein schnelleres Anwachsen der Funktionswerte $f(\lambda_j)$ erreichen, jedoch theoretisch nicht unbedingt eine konvergente Methode. Da es sich ja hier „nur“ um eine Heuristik zur Bestimmung unterer Schranken handelt, ist der Konvergenzaspekt gleichgültig. Man bricht einfach nach einer bestimmten Rechenzeit oder einem STOP-Kriterium ab und benutzt den bisher besten Zielfunktionswert als untere Schranke.

Aus empirischer Erfahrung kann man festhalten, dass die Schrittlängen t_j problemabhängig zu wählen sind und dass sie i. A. durch experimentelle Berechnung bestimmt werden müssen. Ein Beispiel für eine derartige Schrittlängenwahl ist in Schritt 5 des folgenden Algorithmus zu finden. Dieses modifizierte Subgradientenverfahren hat sich bei praktischen Versuchen bezüglich des symmetrischen TSP recht gut bewährt:

Algorithmus (Subgradientenverfahren für symmetrisches TSP).

Eingabe: Symmetrisches TSP mit Zielfunktion c, $c_{ij} \geq 0$, $1 \leq i < j \leq n$.
Ausgabe: Untere Schranke L für den Optimalwert des TSP.

1. Initialisierung:
   $U :=$ obere Schranke für die kürzeste Tour,
   $L := 0$ (gegenwärtig beste untere Schranke),
   $\lambda^T = (\lambda_1, \ldots, \lambda_n) := (0, \ldots, 0)$ (Anfangsknotengewichte),
   $0 < \alpha \leq 2$ (Skalierungsfaktor für die Schrittlänge, i. A. $\alpha := 2$),
   $\varepsilon_1, \varepsilon_2, \varepsilon_3 > 0$ (numerische Genauigkeitsschranken),
   $k \in \mathbb{N}$ (Zählparameter).

2. Berechne den kürzesten 1-Baum B bezüglich c^λ mit $c^\lambda_{ij} := c_{ij} + \lambda_i + \lambda_j \; \forall \, i, j$.
   Falls B eine Tour ist $\longrightarrow$ STOP.

3. Berechne einen Subgradienten u durch $u_i := d_B(i) - 2$ für $i = 1, \ldots, n$.

4. Setze $L := \max\{L, c(B) + \lambda^T u\}$ (gegenwärtig beste untere Schranke).
   - Falls $U - L < \varepsilon_1$ ist $\longrightarrow$ STOP (gewünschte Güte erreicht).
   - Falls sich L in den letzten k Schritten nicht wenigstens um ε_2 verbessert hat $\longrightarrow$ STOP. (Wir geben auf, weil das Verfahren stagniert. Weiterrechnen wäre nur Zeitverschwendung).

5. Setze
   $$t := \alpha \frac{U - L}{\sum_{i=1}^{n} u_i^2} \qquad \text{(Schrittlänge).}$$
   Ist $t < \varepsilon_3 \longrightarrow$ STOP. (Die Schrittlänge ist so klein geworden, dass numerisch nichts mehr passieren wird. Wir geben auf.)

6. Setze $\lambda := \lambda + tu$ und gehe zu Schritt 2. $\triangle$

Die obere Schranke U in Schritt 1 erhält man z. B. durch Anwenden einer Primalheuristik oder man nimmt eine triviale obere Schranke, wie z. B.

$$U = n \cdot \max_{1 \leq i < j \leq n} c_{ij} \qquad \text{oder} \qquad U = \sum_{i=1}^{n} \max_{j \neq i} c_{ij}.$$

Wir haben zu Testzwecken $\alpha := 2$, $k := 100$ und $\varepsilon_1, \varepsilon_2, \varepsilon_3 := 10^{-3}$ gesetzt und für unser 6-Städte Beispiel das obige Programm ausgeführt. Das Subgradientenverfahren hat nach

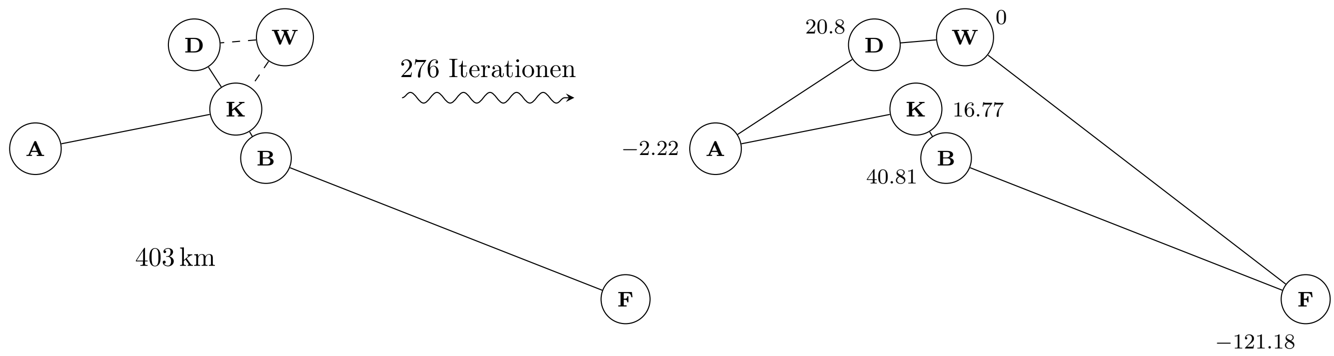

Abbildung 5: Ergebnis des Subgradientenverfahrens für das Rheinland

276 Iterationen (d. h. 276 1-Baum Berechnungen) eine Tour geliefert. Damit ist diese Tour die optimale Lösung unseres 6-Städte Problems. Die optimalen Knotengewichte der 6 Städte sind in Abbildung 5 verzeichnet.

Es ist zu bemerken, dass die Optimallösung von (1.7) im Falle des TSP (und nicht nur hier) nicht unbedingt eine Tour liefern muss, sondern es können durchaus „gaps“ auftreten.

Trotzdem hat sich der Subgradientenansatz über die Lagrange-Relaxierung speziell für Probleme mittlerer Größenordnung bewährt. Verfahren dieser Art liefern i. A. sehr gute untere (bzw. bei Maximierungsproblemen obere) Schranken und kombiniert mit Branch & Bound gehören sie vielfach zu den besten Algorithmen, die wir für schwere kombinatorische Optimierung kennen.

Zum Ende dieses Abschnittes möchten wir noch ein weiteres TSP-Beispiel vorführen und auf einige (numerische) Eigenschaften des Subgradientenverfahrens zur Lösung der Lagrange-Relaxierung hinweisen.

Die Werte (und damit unteren Schranken), die das Subgradientenverfahren liefert, hängen natürlich stark von den gewählten STOP-Kriterien ab. Je nachdem, wie man in dem Spezialverfahren für das TSP die Parameter k, ε_1, ε_2, ε_3, α setzt, wird man natürlich andere Resultate erhalten. Außerdem muss man bei der 1-Baum Relaxierung einen Knoten v spezifizieren. Obwohl theoretisch die Wahl von v irrelevant ist (für alle v ist $\max\{f(\lambda) \mid \lambda \in \mathbb{R}^n\}$ gleich), zeigt sich doch bei praktischen Rechnungen, dass unterschiedliche untere Schranken, abhängig von der Wahl, auftreten können. Wir demonstrieren das am folgenden Beispiel, von dem mir U Zaw Win berichtet hat.

Beispiel. Burma besteht aus 14 Bundesländern. Zur Erinnerung an den Freiheitskampf gegen England und zur Festigung der nationalen Einheit wird in jedem Jahr die Nationalflagge auf eine Rundreise durch die Hauptstädte der 14 Bundesländer geschickt. Diese Hauptstädte sind mit ihren geographischen Koordinaten in Tabelle 3 aufgelistet. Die Rundreise beginnt jeweils am 30. Januar eines jeden Jahres in der Bundeshauptstadt Rangoon, führt dann durch die übrigen 13 Städte und kehrt zum Nationalfeiertag am 12. Februar nach Rangoon zurück. Der Transport der Flagge erfolgt – je nach Verkehrs-

Nr	Stadt	Breitengrad (N)	Längengrad (E)
1	Rangoon	16.47	96.10
2	Bassein	16.47	94.44
3	Sittwe	20.09	92.54
4	Haka	22.39	93.37
5	Myitkyina	25.23	97.24
6	Mandalay	22.00	96.05
7	Taunggyi	20.47	97.02
8	Pegu	17.20	96.29
9	Moulmein	16.30	97.38
10	Tavoy	14.05	98.12
11	Pa-an	16.53	97.38
12	Sagain	21.52	95.59
13	Loikaw	19.41	97.13
14	Magwe	20.09	94.55

Tabelle 3: 14 Städte in Burma (Geografische Koordinaten)

	2	3	4	5	6	7	8	9	10	11	12	13	14
1	153	510	706	966	581	455	70	160	372	157	567	342	398
2		422	664	997	598	507	197	311	479	310	581	417	376
3			289	744	390	437	491	645	880	618	374	455	211
4				491	265	410	664	804	1070	768	259	499	310
5					400	514	902	990	1261	947	418	635	636
6						168	522	634	910	593	18	284	239
7							389	482	757	439	163	124	232
8								154	406	133	508	273	355
9									276	43	623	358	498
10										318	898	633	761
11											582	315	464
12												275	221
13													247

Tabelle 4: 14 Städte in Burma (Distanzen auf der Erdoberfläche)

verbindungen – zu Fuß (Mandalay nach Sagaing), mit dem Schiff, dem Auto oder dem Flugzeug. Wir gehen davon aus, dass die Entfernung zwischen den Städten durch die Luftliniendistanzen auf der Erde gegeben sind, die man aus den Daten der Tabelle 3 berechnen kann; die so bestimmte Entfernungsmatrix findet sich in Tabelle 4. Wie sollte die Rundreise der Nationalflagge ausgelegt werden, so dass der Reiseweg möglichst kurz ist?

Für das Burma-Problem haben wir eine Rundreise mit Hilfe des Farthest-Insert-Algorithmus und dem Anfangskreis 7, 11, 14 berechnet. Diese Rundreise hat die Länge

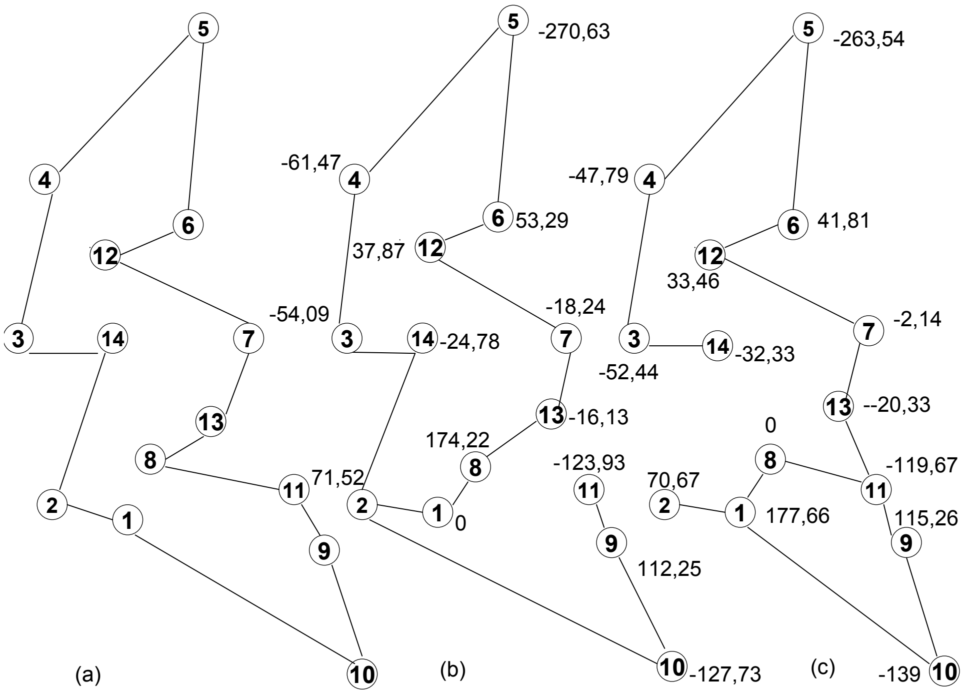

Abbildung 6: Lösungen zum Burma-TSP

3 322 km und ist in Abbildung 6(a) (nicht maßstabsgetreu) dargestellt. Um die Qualität dieser Lösung abzuschätzen, haben wir untere Schranken mit Hilfe des Subgradientenverfahrens berechnet.

Wählen wir als Knoten v des 1-Baumes den Knoten 1 (Rangoon), so erhalten wir den in Abbildung 6(b) gezeigten 1-Baum der Länge 3 313.58398 km. Die „optimalen" Knotengewichte sind in Abbildung 6(b) an den Knoten eingetragen.

Wählen wir als Knoten v den Knoten 8 (Pegu), so erhalten wir einen 1-Baum der Länge 3,316.98 km. Er ist in Abbildung 6(c) dargestellt. Die „optimalen" Knotengewichte sind dort ebenfalls verzeichnet.

Wir sehen also, dass bei unterschiedlicher Wahl des Knotens v in der 1-Baum-Relaxierung aufgrund unserer heuristischen Stopregeln verschiedene untere Schranken berechnet werden können. Da unser Burma-Problem ganzzahlige Daten hat, wissen wir, dass die kürzeste Rundreise nicht kürzer als 3 317 km sein kann. Der maximale absolute Fehler der von uns gefundenen Rundreise beläuft sich somit auf 5 km, dies entspricht einem maximalen relativen Fehler von etwa 0,15 %. $\triangle$