Robustes Clustering von Graphen

12. Juli 2011

Zusammenfassung

Bei vielen praktischen Problemen kann es nötig sein, dass verschiedene Objekte, die jeweils in einer gewissen Relation zueinander stehen, in unterschiedliche Gruppen möglichst sinnvoll aufgeteilt werden sollen. Dieses kann man mathematisch modellieren, indem man einen, den Objekten und den Relationen entsprechenden, Graphen betrachtet und dessen Knotenmenge in disjunkte Mengen zerlegt. Die disjunkten Mengen nennt man in diesem Fall Cluster und die Aufteilung bzw. Zuteilung an sich ist das Clustering. Wie sinnvoll ein Clustering ist hängt jeweils von dem Problem ab, das man betrachtet. Dies resultiert in unterschiedlichen Zielfuntkionen. Außerdem kann der Graph vollständig oder nicht vollständig sein und die Anzahl der Cluster kann vorgegeben sein oder auch nicht. Bei den zwei Clusterproblemen, die im Vortrag betrachtet werden, gehen wir von vollständigen Graphen aus mit einer vorgegebene Anzahl an Clustern. Beim ersten Problem soll der maximale Durchmesser der Cluster minimiert werden und beim zweiten Problem soll der minimale Abstand zwischen den Cluatern maximiert werden. Da die Probleme NP-schwer sind und man daher keine exakte Lösung in Polynomialzeit berechnen kann. ist man auf Heuristiken angewiesen. Die Heuristik die vorgestellt wird, heißt *Hierarchical clustering Algorithmus*. Bei diesem Algorithmus geht man anfangs davon aus, dass alle Knoten in einem eigenen Cluster liegen. Dann werden nach und nach Cluster zusammengefügt, bis die vorgegebene Anzahl an Clustern erreicht wird.
Nach der Vorstellung der Clusterprobleme werden verschiedene Robustheitskonzepte vorgestellt. Dabei wird hauptsächlich auf die strenge Robustheit und die Abweichungsrobustheit eingegangen. Im letzten Abschnitt des Vortrags wird anfangs das streng robuste Clusteringproblem vorgestellt. Wie sich letztendlich zeigen wird, läßt sich die streng robuste Version auf ein einfaches Clusteringproblem reduzieren, indem man für jede Kante den größtmöglichen bzw. kleinstmöglichen annehmbaren Wert nimmt. Zur Abweichungsrobustheit werden dann noch erste Ideen zur Vorgehensweise und kleinere Erkenntnisse präsentiert.

Notation

- $X \mathrel{\hat{=}}$ Menge aller möglichen Cluster x, die aus p Partitionen bestehen.
- $C(i) :=$ Das Cluster, welches der Knoten i zugeordnet ist
- $D_x^1 := \{\{i,j\} \mid C(i) = C(j)\} \mathrel{\hat{=}}$ Menge der Kanten, die beim Clustering x innerhalb der Cluster verlaufen.
- $D_x^2 := \{\{i,j\} \mid C(i) \neq C(j)\} \mathrel{\hat{=}}$ Menge der Kanten, die beim Clustering x zwischen den Cluster verlaufen.
- $\overline{c_x} := \max\{c_{ij} \mid \{i,j\} \in D_x\}$
- $\underline{c_x} := \min\{c_{ij} \mid \{i,j\} \in D_x\}$
- $\overline{c_x^g} := \max\{g(c_{ij}) \mid \{i,j\} \in D_x\}$
- $\underline{c_x^g} := \min\{g(c_{ij}) \mid \{i,j\} \in D_x\}$
- $\overline{c_{ij}} := \sup\{c_{ij} \mid c \in \mathcal{U}\}$
- $\underline{c_{ij}} := \inf\{c_{ij} \mid c \in \mathcal{U}\}$

- $\overline{c} : E \to \mathbb{R};\ c(\{i,j\}) = \overline{c_{ij}}$
- $\underline{c} : E \to \mathbb{R};\ c(\{i,j\}) = \underline{c_{ij}}$
- $\overline{\overline{c_x}} := \max\{\overline{c_{ij}} \mid \{i,j\} \in D_x\}$
- $\underline{\underline{c_x}} := \min\{\underline{c_{ij}} \mid \{i,j\} \in D_x\}$

Clustering von Graphen

Clusteringproblem 1 (Complete-link Clustering)

Instanz: Vollständiger Graph $G = (V, E)$, Kantengewichte $c : E \to \mathbb{R}$ und die Anzahl der Cluster p.
Aufgabe: Finde ein Clustering x welches die maximale Kosten/Entfernung innerhalb der Cluster minimiert.
Zielfunktion: $f : X \to \mathbb{R}\ ;\ f(x) := \max\{c_{ij} \mid \{i,j\} \in D_x^1\}$
Optimierungsproblem: $\min\{f(x) \mid x \in X\}$

Clusteringproblem 2 (Single-link Clustering)

Instanz: Vollständiger Graph $G = (V, E)$, Kantengewichte $c : E \to \mathbb{R}$ und die Anzahl der Cluster p.
Aufgabe: Finde ein Clustering x welches die minimale Kosten/Entfernung zwischen den Cluster maximiert.
Zielfunktion: $f : X \to \mathbb{R}\ ;\ f(x) := \min\{c_{ij} \mid \{i,j\} \in D_x^2\}$
Optimierungsproblem: $\max\{f(x) \mid x \in X\}$

Formulierung als IP:

$$x_{i,s} = \begin{cases} 1 & \text{Knoten } i \text{ ist in Cluster } s \\ 0 & \text{sonst} \end{cases}$$

Clusterproblem 1 (Complete-link Clustering):

$$\begin{aligned} \min\ & z \\ s.d.\ & z \geq c_{ij} \cdot (x_{is} + x_{js} - 1)\ \ \forall i,j \text{ mit } i \neq j,\ \ \forall s \in \{1,...,p\} \\ & \sum_{s=1}^{n} x_{is} = 1\ \ \forall i \in \{1,...,n\} \\ & z \in \mathbb{R},\ \ x_{is} \in \mathbb{B}\ \ \forall i\ \forall s \end{aligned}$$

Clusterproblem 2 (Single-link Clustering):

$$
\begin{aligned}
\max\ & z \\
s.d.\ & z \leq c_{ij} \cdot \sum_{s,r:s\neq r} x_{is}x_{jr} + M \cdot \sum_{s=1}^{p} x_{is}x_{js} \quad \forall i,j \text{ mit } i \neq j \\
& \sum_{s=1}^{n} x_{is} = 1 \quad \forall i \in \{1,...,n\} \\
& M \geq c_{ij} \quad \forall i,j \text{ mit } i \neq j \\
& z \in \mathbb{R}, \quad M \in \mathbb{R}, \quad x_{is} \in \mathbb{B} \quad \forall i\ \forall s
\end{aligned}
$$

Hierarchical clustering Algorithmus (Complete-link):

1. Erstelle eine $n \times n$-Matrix A mit Einträgen $a_{ij} = c_{ij}$ für $i \neq j$ und $a_{ii} = 0$. Setze $m = 0$.
2. Finde das Clusterpaar $(r), (s)$ mit dem kleinsten Abstand: $c[(s),(r)] = \min c[(i),(j)]$.
3. Ist $m = n - p$ STOPP. Ansonsten weiter zu Punkt 4.
4. Vereinige die Cluster (r) und (s) zu einem Cluster (r,s).
5. Passe die Matrix A an, indem die Spalten und Zeilen, die zu den Clustern (r) und (s) gehören, gelöscht werden und eine neue Spalte und Zeile hinzugefügt wird, die zum Cluster (r,s) gehört. Für alle anderen Cluster (k) setze dann $c[(k),(r,s)] = \max\{c[(k),(s)], c[(k),(r)]\}$.
6. Setze $m = m + 1$. Zurück zu Punkt 2.

Satz 1 *Sei $G = (V,E)$ ein vollständiger Graph mit Kantengewichten c und somit (G,c) eine Instanz eines Clusteringproblems. Sei g eine streng monoton steigende Funktion. Dann ist $(G, g(c))$ eine Instanz eines Clusteringproblems auf das (G,c) in Polynomialzeit reduzierbar ist.* □

Beweis (Complete-link) Die Berechnung der $g(c_{ij})$ erfolgt in polynomialer Zeit.
Es ist $\overline{c_x^g} = \max\{g(c_{ij}) \mid \{i,j\} \in D_x\} \overset{g \text{ streng monoton}}{\Longrightarrow} g(\max\{c_{ij} \mid \{i,j\} \in D_x\}) = g(\overline{c_x})$.

Sei x^* Lösung von (G,c). Annahme: x^* ist nicht optimal für $(G, g(c))$. Dann gibt es Lösung y mit $\overline{c_y^g} < \overline{c_x^g} \Longrightarrow g(\overline{c_y}) < g(\overline{c_{x^*}}) \overset{g \text{ streng monoton}}{\Longrightarrow} \overline{c_y} < \overline{c_x^*}$. ↯dazu, dass x^* optimal für (G,c) ist.

Sei x^* Lösung von $(G, g(c))$. Annahme: x^* ist nicht optimal für (G,c). Dann gibt es Lösung y mit $\overline{c_y} < \overline{c_{x^*}} \overset{g \text{ streng monoton}}{\Longrightarrow} g(\overline{c_y}) < g(\overline{c_{x^*}}) \Longrightarrow \overline{c_y^g} < \overline{c_x^g}$. ↯dazu, dass x^* optimal für $(G, g(c))$ ist.

Insgesamt folgt: x^* ist optimale Lösung von $(G,c) \Longleftrightarrow x^*$ ist optimale Lösung von $(G, g(c))$.

Robustheitskonzepte

Strenge Robustheit

Definition 1 Sei $P(c), c \in \mathcal{U}$ ein unsicheres Optimierungsproblem. Die (streng) robuste Version (RC) ist gegeben durch:

$$
\begin{aligned}
\min\ & \sup_{c \in \mathcal{U}} f(x,c) \\
s.d.\ & F(x,c) \leq 0 \quad \forall c \in \mathcal{U}
\end{aligned}
$$

Definition 2 Sei x eine streng robuste Lösung. Ein worst-case Szenario $c^w(x)$ bzgl. x ist ein $c \in \mathcal{U}$, dass $f(x, c)$ maximiert. □

Abweichungsrobust

Definition 3 Ein Abweichungs-worst-case Szenario $c^{aw}(x)$ bzgl. x ist ein $c \in \mathcal{U}$, das $f(x, c) - f(x^*(c), c) = f(x, c) - f^*(c)$ maximiert.
Die *robuste Abweichung* von x ist dann $abw(x) := \sup_{c \in \mathcal{U}}(f(x, c) - f^*(c))$. □

Definition 4 Eine abweichungsrobuste optimale Lösung ist eine streng robuste Lösung x, die $abw(x)$ minimiert. □

Robustes Clustering

Streng robustes Clustering

Satz 2 (Complete-Link) *Es sei $G = (V, E)$ ein vollständiger Graph mit unsicheren Kosten c und $\mathcal{U}$ die dazugehörige Unsicherheitsmenge. Die streng robuste Version des unsicheren Optimierungsproblems $P(c), c \in \mathcal{U} : \min \sup_{c \in \mathcal{U}} f(x, c),\ s.d. x \in X$ ist dann äquivalent zum Optimierungsproblem $P(\overline{c})$:*

$$\begin{aligned} &\min\ f(x, \overline{c}) \\ &s.d.\ x \in X \end{aligned}$$

Beweis Sei $x \in X$ beliebig. Dann ist

$$\sup_{c \in \mathcal{U}} f(x, c) = \sup_{c \in \mathcal{U}} \max\{c_{ij} \mid \{i, j\} \in D_x^1\} = \sup_{c \in \mathcal{U}} \max_{\{i,j\} \in D_x^1} c_{ij} =$$
$$\max_{\{i,j\} \in D_x^1} \sup_{c \in \mathcal{U}} c_{ij} = \max\{\overline{c_{ij}} \mid \{i, j\} \in D_x^1\} = \max\{\overline{c_{ij}} \mid \{i, j\} \in D_x^1\} = f(x, \overline{c})$$

Für jedes beliebege $x \in X$ liefert daher $\sup_{c \in \mathcal{U}} f(x, c)$ und $f(x, \overline{c})$ denselben Zielfunktionswert. Da auch die zulässige Menge identisch ist, sind die Probleme äquivalent. ■