4. Aufgabenblatt zur Vorlesung

Höhere Algorithmik

WiSe 2016/17

Wolfgang Mulzer, Boris Klemz

Abgabe bis zum 18. November 2016, 12 Uhr, in den Briefkasten neben Raum 111

Aufgabe 1 Varianten der Vorlesungsbeispiele *10 Punkte*

(a) Betrachten Sie die Variante des Einkaufsproblems, bei dem von jedem Artikel beliebig viele Exemplare zur Verfügung stehen. Das heißt, wir haben Artikel 1 bis n gegeben und jeder Artikel hat einen Preis p_i und einen Wert w_i. Wir haben ein Budget B und möchten eine *Multimenge* von Artikeln finden, die unser Budget beachtet und den Wert maximiert.

Zeigen Sie, wie man auch diese Variante des Einkaufsproblems in Zeit $O(nB)$ lösen kann.

(b) In der Vorlesung haben Sie gesehen, wie das Rundreiseproblem mit Hilfe von dynamischem Programmieren gelöst werden kann. Arbeiten Sie die Details des Algorithmus aus und geben Sie Pseudocode an, um eine optimale Tour zu berechnen.

Aufgabe 2 Matrizenkettenmultiplikation *10 Punkte*

Für eine gegebene Folge M_1, M_2, …, M_n von n Matrizen ist das *Matrizenkettenprodukt* $M_1 \cdot M_2 \cdot \cdots \cdot M_n$ zu berechnen. Die Matrizen haben dabei verschiedene Dimensionen. M_1 ist eine $p_1 \times p_2$ Matrix, M_2 ist eine $p_2 \times p_3$ Matrix, usw.

Um eine $a \times b$ Matrix mit einer $b \times c$ Matrix (naiv) zu multiplizieren, benötigen wir bekanntlich acb Multiplikationen und $ac(b-1)$ Additionen, also insgesamt $ac(2b-1)$ elementare Operationen. Daraus folgt, dass es einen Unterschied macht, wie man das Matrizenprodukt klammert, also in welcher Reihenfolge man die Matrizen multipliziert.

Sei beispielsweise M_1 eine 100×200 Matrix, M_2 eine 200×10 Matrix und M_3 eine 10×1 Matrix. Dann haben wir zwei Möglichkeiten, die Multiplikation durchzuführen: $(M_1M_2)M_3$ oder $M_1(M_2M_3)$. Im ersten Fall benötigen wir zunächst $100 \cdot 10 \cdot (400-1) = 399.000$ Operationen, um die 100×10 Matrix M_1M_2 zu berechnen, und dann $100 \cdot 1 \cdot (20-1) = 1.900$ Operationen, um das Ergebnis zu erhalten. Insgesamt ergibt das 400.900 Operationen. Im zweiten Fall brauchen wir erst $200 \cdot 1 \cdot (20-1) = 3.800$ Operationen für die 200×1 Matrix M_2M_3 und dann $100 \times 1 \times (400-1) = 39.900$ Operationen für das Ergebnis. Insgesamt sind es hier 43.700 Operationen. Die zweite Klammerung ist also fast zehnmal so schnell wie die erste. Ziel dieser Aufgabe ist es, eine optimale Klammerung für eine Matrizenkettenmultiplikation zu bestimmen.

(a) Bezeichne mit $P[i, j]$ die Kosten für eine optimale Klammerung des Matrizenkettenprodukts $M_i \cdot \cdots \cdot M_j$ $(1 \leq i \leq j \leq n)$. Unser Ziel ist, $P[1, n]$ zu berechnen. Finden Sie eine geeignete Rekursionsgleichung für $P[i, j]$.

(b) Benutzen Sie Ihre Rekursion, um Pseudocode für einen Algorithmus anzugeben, welcher die optimalen Kosten $P[1, n]$ bestimmt. Analysieren Sie Laufzeit und Platzbedarf.

(c) Erweitern Sie Ihren Algorithmus so, dass er auch eine optimale Klammerung ausgibt.

(d) (*freiwillig, 5 Zusatzpunkte*) Implementieren Sie ein Programm, das eine optimale Klammerung für eine gegebene Matrizenkettenmultiplikation bestimmt und die Multiplikation durchführt. Vergleichen Sie Ihr Programm mit einer naiven Implementierung, welche die Matrizen von links nach rechts multipliziert.

Aufgabe 3 Versteckte Markov-Modelle *10 Punkte*

(a) Betrachten Sie das folgende versteckte Markov-Modell.

- Zustände: $Q = \{q, r, s\}$.
- Alphabet: $\Sigma = \{a, b\}$.
- Anfangsverteilung: $\{q : 0.1,\ r : 0.4,\ s : 0.5\}$.
- Ausgabeverteilung für q: $\{a : 0.2,\ b : 0.8\}$.
- Ausgabeverteilung für r: $\{a : 0.7,\ b : 0.3\}$.
- Ausgabeverteilung für s: $\{a : 0.5,\ b : 0.5\}$.
- Übergangsverteilung für q: $\{q : 0.8,\ r : 0.1,\ s : 0.1\}$.
- Übergangsverteilung für r: $\{q : 0.3,\ r : 0.3,\ s : 0.4\}$.
- Übergangsverteilung für s: $\{q : 0.2,\ r : 0.4,\ s : 0.4\}$.

Benutzen Sie den Viterbi-Algorithmus, um die wahrscheinlichste Erklärung für die Ausgabefolge *abba* zu ermitteln.

(b) Bei einer Implementierung des Viterbi-Algorithmus rechnet man oft mit den Werten $\log p_i$ statt den Wahrscheinlichkeiten p_i. Erklären Sie, warum das eine gute Idee ist.

(c) Eine Anwendung von versteckten Markov-Modellen ist die Fehlerkorrektur. Wir wollen uns an einem Beispiel überlegen, wie dies prinzipiell funktionieren kann.

Angenommen, wir möchten einen deutschsprachigen Text über dem Alphabet $\{a, b, c, \ldots, z, ' '\}$ über eine gestörte Leitung übermitteln ($' '$ stellt das Leerzeichen dar, der Einfachheit halber werden Satzzeichen ignoriert). Wir wissen, dass die Leitung ungefähr 10 Prozent der übertragenen Zeichen zufällig ändert. Trotzdem möchten wir aus der empfangenen Nachricht einen möglichst korrekten Text rekonstruieren.

Beschreiben Sie, wie man diese Situation mit Hilfe eines versteckten Markov-Modells modellieren kann. Was sind die Zustände? Wie könnte man die zugrunde liegenden Verteilungen sinnvoll ermitteln? Welche Annahmen werden in dem Modell gemacht? Inwiefern treffen diese Annahmen in der Wirklichkeit zu?