Theoretische Informatik 1 / Algorithmentheorie

Wintersemester 2012/13

Prof. Dr. Georg Schnitger,
Dipl. Inf. Bert Besser,
Dipl. Inf. Matthias Poloczek

Arbeitsgruppe Theoretische Informatik, Institut für Informatik

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Blatt 8

Ausgabe: 20.12.2012
Abgabe: **17**.01.2013 **vor** der Vorlesung

Wir wünschen allen Teilnehmern
ein frohes Weihnachtsfest
und einen guten Rutsch
ins neue Jahr!

8.1. Aufgabe (8) *Textsatz*

Wir möchten einen Text bestehend aus n Worten im Blocksatz setzen. Das heißt, der Text mit Ausnahme der letzten Zeile wird links- wie rechtsbündig dargestellt. Der Text besteht aus den Worten $w_1, \ldots, w_n$. Eine Zeile bietet Platz für L Zeichen (inlusive Leerzeichen). Wir nehmen $|w_i| \leq L$ für alle i an.

Beim Setzen des Textes darf die Reihenfolge der Worte naheliegenderweise nicht verändert werden und wir wollen auf Silbentrennung verzichten. Also bleibt uns nur die Möglichkeit, in einzelnen Zeilen etwas weniger als L Zeichen unterzubringen. Davon wollen wir nur sparsam Gebrauch machen, um den Text möglichst gleichförmig und lesbar zu gestalten.

Um die Lesbarkeit einer Zeile zu beschreiben, ist eine Funktion `ugly()` entwickelt worden. Sie erhält als Eingabe eine Folge von Worten und eine Zeilenlänge und liefert als Ergebnis eine Zahl, die umso höher ist, je hässlicher die Worte in einer Zeile der Länge L aussehen. (Der Wert 0 bedeutet, *keine Beanstandungen.* Je höher das Ergebnis, umso verrenkter die Zeile.) Wir nehmen weiter an, dass für die letzte Zeile nur gefordert ist, dass sie nicht mehr als L Zeichen enthält – dann ist ihr `ugly()` Wert 0.

Die Aufgabe ist nun, Zeilenumbrüche zu finden, so dass die Summe der `ugly()` Werte aller Zeilen minimiert wird. **Entwirf** einen Algorithmus, der dieses Problem in Zeit $O(n^2)$ löst.

Hinweis: Dynamische Programmierung. Die Funktion `ugly()` kann für alle Wortfolgen $w_i, \ldots, w_j$ mit $i < j$ und Parameter L aufgerufen werden. Ihre Laufzeit darf als konstant unterstellt werden.

Übrigens: Ein **mögliches** Beispiel für `ugly()` wäre:

$$\texttt{ugly}(w_i, \ldots, w_j, L) = \begin{cases} \left(L - j + i - \sum_{k=i}^{j} |w_k|\right)^2 & j - i + \sum_{k=i}^{j} |w_k| \leq |L| \\ \infty & \text{sonst} \end{cases}$$

8.2. Aufgabe (8) *Multiprocessor Scheduling*

Im Problem des *Multiprocessor Scheduling* erhalten wir als Eingabe:

- eine Menge von n Aufgaben, wobei die i-te Aufgabe die Bearbeitungszeit $b(i)$ besitzt,
- die Anzahl m der zur Verfügung stehenden Maschinen sowie
- eine Frist F.

Es ist zu entscheiden, ob die n Aufgaben auf die m Maschinen verteilt werden können, sodass die Frist F eingehalten wird. (Die Frist F wird eingehalten, wenn jede Maschine nur Aufgaben mit Gesamtbearbeitungszeit höchstens F erhält. Alle Maschinen sind identisch und insbesondere hängt $b(i)$ nicht von der jeweiligen Maschine ab.)

a.) **Zeige**, dass das Multiprocessor Scheduling Problem in NP liegt.

b.) **Weise nach**, dass Multiprocessor Scheduling NP-vollständig ist.

Hinweis: Du kannst annehmen, dass das Partition Problem NP-vollständig ist. Im *Partition* Problem sind k natürliche Zahlen $a_1, ..., a_k$ gegeben, und es ist zu entscheiden, ob die k Zahlen in zwei Klassen mit gleicher Gesamtsumme aufgeteilt werden können, also:

$(a_1, ..., a_k) \in$ Partition $\Leftrightarrow$ Es gibt eine Indexmenge $I \subseteq \{1, \ldots, k\}$ mit

$$\sum_{i \in I} a_i = \sum_{j \in \{1,2,...,k\} \setminus I} a_j.$$

8.3. Aufgabe (8) *Teilgrapheneinbettung (GE)*

Die Sprache GE besteht aus allen Paaren

$$(G_1, G_2)$$

von ungerichteten Graphen $G_1 = (V_1, E_1)$ und $G_2 = (V_2, E_2)$, sodass G_1 als Teilgraph in G_2 eingebettet werden kann.
Wir sagen, dass G_1 in G_2 eingebettet werden kann, wenn es eine injektive Abbildung $f : V_1 \to V_2$ mit der Eigenschaft

$$\text{für alle Kanten } \{u, v\} \in E_1 \text{ gilt: } \{f(u), f(v)\} \in E_2$$

gibt. **Beweise** folgende Aussagen:

a) $GE \in$ NP

b) $CLIQUE \leq_p GE$