Abteilung Algorithmik
Institut für Betriebssysteme und Rechnerverbund **SS 08**
TU Braunschweig

Prof. Dr. Sándor Fekete
Christiane Schmidt

Online-Algorithmen
Übung 5 vom 23.06.2008

Abgabe der Lösungen am Montag, den 14.07.08, vor der Übung im IZ 262A.
Bitte die Blätter vorne deutlich mit eigenem Namen versehen!

Aufgabe 1 (Load balancing): In der Übung haben wir das Problem der Lastbalancierung für identische Maschinen und permanente Jobs betrachtet. Dabei haben wir gezeigt, dass der GREEDY-Algorithmus, der den Job jeweils der Maschine mit geringster Last vor Zuweisung zuweist, $(2 - \frac{1}{m})$-kompetitiv ist.
Zeige, dass die dort etablierte obere Schranke auch für den Fall von Jobs mit endlicher (bekannter oder unbekannter) Dauer gilt.

(10 Punkte)

Aufgabe 2 (Scheduling):
Im Folgenden kommen n Jobs $J_1, \ldots, J_n$ an, wir haben aber zum Einen nur eine Maschine zur Verfügung, zum Anderen gibt es für die Jobs weitere Einschränkungen:

- die Maschine kann zu jedem Zeitpunkt nur genau einen Job bearbeiten,
- jeder Job J_j muss ununterbrochen für die Dauer $p_j > 0$ auf der Maschine bearbeitet werden,
- jeder Job kann erst ab einem Zeitpunkt r_j bearbeitet werden,
- jeder Job hat ein Fälligkeitsdatum d_j.

Ziel ist es, die Jobs so der Maschine zuzuweisen (einen Schedule Σ zu bestimmen), dass die maximale *Verspätung* über alle Jobs minimiert wird. Der Schedule bestimmt die Anfangszeiten $s_1, \ldots, s_n$ für die Jobs. Die Verspätung L_j eines Jobs ist dabei gegeben als Differenz zwischen der (von der gewählten Heuristik abhängigen) Fertigstellungszeit C_j und dem Fälligkeitsdatum:

$$L_j(\Sigma) = C_j(\Sigma) - d_j. \tag{1}$$

Die maximale Verspätung ergibt sich also zu: $L_{max} = \max_{1 \leq j \leq n} L_j$.
Die maximale Verspätung kann in diesem Modell jedoch negativ werden. Aber nichtpositive Fälligkeitsdaten würden eine unrealistsiche Anforderung darstellen. Ein weiteres Model führt daher Lieferzeiten ein. Jeder Job hat eine Lieferzeit q_j: nachdem er auf der Maschine bearbeitet wurde wird er erst nach einer zusätzlichen Zeit von q_j (z.B. auf einer weiteren Maschine, die keinen Engpass darstellt) ausgeliefert. Nun können verschiedene Lieferungen zeitlich überlappen . Der Lieferfertigstellungszeitpunkt von J_j ist $s_j + p_j + q_j$

(wollen wir Instanzen aus dem obigen Modell integrieren, setzen wir $q_j = -d_j$). Sei L^*_{max} die minimale maximale Verspätung über alle schedules.
Es gilt:

$$L_{max} = \max_{1 \leq j \leq n} s_j + p_j + q_j \tag{2}$$

Sei im Weiteren $L_j = C_j + q_j$ die Lieferfertigstellungszeit von J_j.
Es gelten:

$$L^*_{max} \geq P = \sum_{j=1}^{n} p_j \tag{3}$$

$$L^*_{max} \geq r_j + p_j + q_j \tag{4}$$

Betrachte Grahams Algorithmus **List Scheduling(LS)** (analog der GREEDY-Strategie für Lastbalancierung): Wenn die (eine) Maschine frei wird, weise ihr den ersten verfügbaren Job zu. Ein Job ist verfügbar, nachdem er für die Bearbeitung freigegeben wurde.

(a) Begründe die Schranken (3) und (4).

(b) Zeige: $L^{LS}_{max} < 2L^*_{max}$.

(20 Punkte)

Aufgabe 3 (k-Server-Problem und der DOUBLE-COVERAGE Algorithmus): In Aufgabe 2 auf Blatt 2 haben wir gesehen, dass der GREEDY-Algorithmus für das k-Server-Problem nicht notwendigerweise kompetitiv ist. Wir betrachten daher den Algorithmus DOUBLE-COVERAGE für Server auf einer Linie:

DOUBLE-COVERAGE

- Wenn die Anfrage außerhalb der konvexen Hülle der Server liegt, bediene sie mit dem nächstgelegenen Server.
- Andernfalls liegt die Anfrage zwischen zwei Servern. Bewege dann beide Server mit gleicher Geschwindigkeit auf die Anfrage zu, bis (mindestens) einer der Server den Anfragepunkt erreicht.

Wieso wird dieser Algorithmus für das für den GREEDY-Algorithmus betrachtete Beispiel nicht beliebig schlecht?

(10 Punkte)

Aufgabe 4 (GREEDY-ONLINE): In der Vorlesung wird der GREEDY-ONLINE Algorithmus für die online Exploration eines einfachen, rektilinearen Polygons durch einen Roboter mit kontinuierlicher Sicht vorgestellt. Betrachte das Polygon in Abbildung 1, mit Startpunkt in s.
Zeichne den Weg des Algorithmus ein. Gib dabei jeweils die extensions und die zugehörigen Reflexknoten oder blockierenden Ecken an! (Für nach dem "taut thread"-Prinzip gebildete Teilwege sollte der L_2-Weg angegeben werden.)

(20 Punkte)

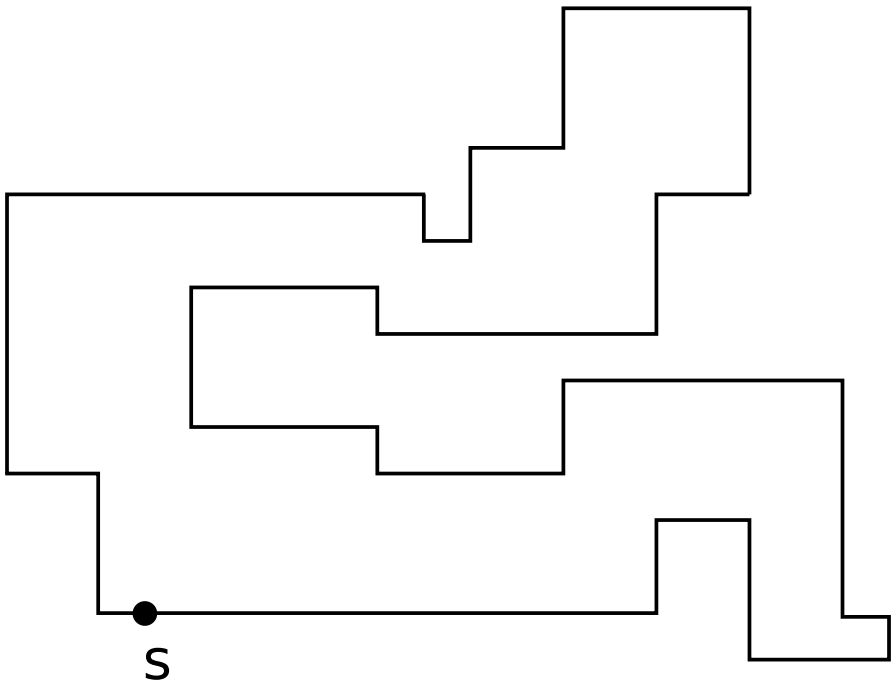

Abbildung 1: Einfaches, rektilineares Polygon mit Startpunkt s.