OPERATIONS RESEARCH
KACKERTSTR. 7
52072 AACHEN

Prof. Dr. Marco Lübbecke

Michael Bastubbe

Praktische Optimierung mit Modellierungssprachen
Aufgabenblatt 3

Abgabe (für eine 1,0): bis spätestens Montag, den 12.06.2015 um 23:59 Uhr,
über das Online-Abgabesystem

Aufgabe 3 (Losgrößenplanung)

In dieser Programmieraufgabe soll das Problem der Losgrößenplanung (*lot sizing*) betrachtet werden. Gegeben sei eine Maschine, die n verschiedene Produkte $j = 1, \ldots, n$ herstellen kann, ein Lager, in dem (der Einfachheit halber) eine beliebige Menge an Produkteinheiten gelagert werden kann, sowie ein Zeithorizont von $T + 1$ Zeitperioden $t = 0, 1, \ldots, T$.

Wir nehmen an, dass von jedem Produkt j zu Beginn (definiert als das Ende der Periode $t = 0$) bereits l_{j0} Einheiten auf Lager sind. Die Maschine beginnt ab Periode $t = 1$ zu produzieren. Dabei muss in jeder Zeitperiode $t \geq 1$ und für jedes Produkt j mit Hilfe der produzierten Menge sowie evtl. aus der Vorperiode gelagerten Einheiten ein Kundenbedarf von d_{jt} Einheiten gedeckt werden. Überschüssige Einheiten werden wiederum gelagert und stehen in der nächsten Periode zur Verfügung; dabei fallen jedoch Lagerhaltungskosten von h_j (in €) pro Einheit an.

In jeder Periode $1 \leq t \leq T$ steht die Maschine für insgesamt K_t Stunden zur Verfügung. Für die Herstellung von Produkt j werden pro Einheit a_j Stunden aufgewendet (der sog. *Produktionskoeffizient*).

Die Maschine ist jedoch immer nur für ein Produkt *gerüstet*, d. h. sie kann nur ein Produkt gleichzeitig herstellen. Es ist allerdings möglich, die Maschine bis zu einmal pro Zeitperiode auf ein anderes Produkt umzurüsten. Somit können pro Periode zwei verschiedene Produkte hergestellt werden: Das Produkt auf das die Maschine am Ende der letzten Periode gerüstet war, sowie das Produkt auf das sie ggf. während der aktuellen Periode gerüstet wurde. Auf welches Produkt die Maschine zu Beginn ($t = 0$) gerüstet ist, kann frei entschieden werden.

Bei der Umrüstung der Maschine fallen zudem Umrüstkosten von s_{ij} (in €) an, wenn die Maschine von Produkt i auf Produkt j umgerüstet wird. Darüber hinaus dauert das Umrüsten st_{ij} Stunden, während dieser die Maschine nicht für die Produktion zur Verfügung steht.

Gesucht ist nun ein Produktionsplan, der entscheidet, von welchem Produkt pro Periode jeweils welche Mengen hergestellt und gelagert werden, sodass die anfallenden Gesamtkosten – Summe aus Lagerhaltungs- und Umrüstkosten – minimal werden.

Im Lernraum findet Ihr die Datensätze **lotData1.txt**, **lotData2.txt** und **lotData3.txt**, die jeweils wie folgt aufgebaut sind:

```
nProducts ...
Timehorizon ...
l: ...
h: ...
```

```
K: ...
a: ...
d1: ...
d2: ...
.
.
.
s1: ...
s2: ...
.
.
.
st1: ...
st2: ...
.
.
.
```

Hierbei gibt `nProducts` die Anzahl n der Produkte an, `Timehorizon` steht für die Anzahl T der Zeitperioden (ohne Periode 0). Hinter `l:` stehen nacheinander die Lageranfangsbestände l_{j0} (in Spalte $j+1$ steht l_{j0}). Analog dazu beschreiben `h`, `K` und `a` die Lagerhaltungskosten h_j, die Maschinenkapazitäten K_t bzw. die Produktionskoeffienten a_j. Schließlich bezeichnen analog die Zeilen beginnend mit `di`, `si` und `sti` (für Produkt i) die Nachfragen pro Periode sowie die Umrüstkosten bzw. -zeiten von Produkt i auf das Produkt j.

Entwickelt ein IP, welches das beschriebene Problem modelliert. Entwickelt python-Code, der die Instanzen einlesen und euer IP als gurobi-Modell baut. Hochzuladen ist Code, der Instanzdateien einlesen kann, das gurobi-Modell baut und zurückgibt mit dem ihr das Problem gelöst habt. Genauer gesagt ein Zip-archiv, das (mindestens) eine Datei `runlotsizing.py` enthält, in der eine Methode `solve(full_path_instance)` implementiert ist, die euer gurobi-Modellobjekt **`zurückgibt!`**, wobei `full_path_instance` der Pfad zur Instanzdatei (als String) ist. Dabei ist weiterhin zu beachten:

- in den Instanzdateien kann sich die Reihenfolge der Zeilen ändern, euer Code sollte damit aber umgehen können
- das Modell soll mindestens die folgenden Variablen beinhalten, deren Namen folgendermaßen gewählt werden müssen: "x_i_t" wobei $i \in \{1, \ldots, n\}$ (Indizes der Produkte) und $t \in \{1, \ldots, T\}$ (Zeitperioden ohne 0-te Periode) sind, dabei soll x_i_t genau der Menge entsprechen, die von Produkt i in Periode t produziert wird
- stellt sicher, dass die Dateien im Archiv ausreichen um durch Aufruf der Methode `solve(full_path_instance)` die Instanz einzulesen und euer Modell zu returnen

Als optimale Zielfunktionswerte solltet Ihr 39, 19822 bzw. 33719 erhalten. Bitte ladet Euer Archiv über das Online-Abgabesystem bis spätestens Montag, den 12.06., um 23:59 Uhr hoch. Diese Aufgabe wird in der Woche ab dem 12.06. testiert. Weitere Infos folgen im Lernraum.

Viel Erfolg!