Thomas Rothvoß
Location: ELD120 Discussion: 21.04.10

Exercises

Approximation Algorithms

Spring 2010

Sheet 7

Exercise 1

The following is called the SANTA CLAUS PROBLEM: Santa Claus has presents $1,\dots,n$, that he wants to distribute among children $1,\dots,m$, where p_{ij} is the value that kid i has for present j. Santa's goal is that the least luckiest kid is as happy as possible, that means he tries to achieve

$$OPT = \max_{I_1\dot\cup\dots\dot\cup I_m=\{1,\dots,n\}} \left\{ \min_{i=1,\dots,m} \left\{ \sum_{j\in I_i} p_{ij} \right\}\right\}$$

Suppose you know the value of OPT. Let $p_{\max} := \max_{i=1,\dots,m} \max_{j=1,\dots,n} p_{ij}$. Give a polynomial time algorithm that assigns the presents to children such that the happiness of every child is at least $OPT - p_{\max}$. Does this give you any approximation factor?

Hint: Probably you already noticed that this problem has much in common with the UNRELATED MACHINE SCHEDULING problem from the lecture.

Solution:

We use variable x_{ij} to decide, whether to assign present j to child i. Let J be the set of presents and M be the set of children.

(1) Compute basic solution x^* to

$$\begin{aligned} \sum_{i\in M} x_{ij} &= 1 \quad \forall j \in J \\ \sum_{j\in J} p_{ij}x_{ij} &\geq OPT \quad \forall i \in M \\ x_{ij} &\geq 0 \quad \forall i \in M\ \forall j \in J \end{aligned}$$

(2) $x^*_{ij} = 1 \Rightarrow$ assign present j to child i

(3) For not yet assigned presents: Assign present j to a child i with $0 < x^*_{ij} < 1$ s.t. every child receives at most 1 present less

Let us now argue, how step (3) is possible. We consider again the bipartite graph $H = (J \cup M, E)$ with $E = \{(j,i) \mid 0 < x^*_{ij} < 1\}$. Let $LP(\overline{E})$ be the LP that we obtain by freezing variables not in $\overline{E}$. Then the induced solution $\overline{x}^* = (x^*_{ij})_{(j,i)\in\overline{E}}$ is still a basic solution of that LP. $LP(\overline{E})$ has $|\overline{J}| + |\overline{M}|$ constraints, hence

$$|\overline{E}| = |\{(j,i) \mid 0 < x^*_{ij} < 1\}| \leq |\overline{J}| + |\overline{M}|$$

Since $\overline{E}$ is a connected subgraph, it must be a tree plus at most one more edge.

Do the following: Remove all childs that are leaves (then the child gets 1 present less than in OPT). If then a present becomes a leaf then give the present to the father (which is a child) and remove the present.

We are left with an even length child-present cycle. Now take one of the 2 perfect matchings in it and assign the presents to the matched childs. In such a way, we take a way at most 1 present from each child. More precisely consider any child $i \in M$. From the set of presents j with $0 < x^*_{ij} < 1$, the child receives all but at most one present completely.

Unfortunately this algorithm does not guarantee any approximation factor since one may have $p_{ij} = OPT$ for some pairs i, j, then some children might get almost $OPT - p_{ij} \approx 0$.

Exercise 2

Recall that the following problem is **NP**-hard:

> 3-DIM MATCHING: Given disjoint sets $A = \{a_1, \dots, a_n\}, B = \{b_1, \dots, b_n\}, C = \{c_1, \dots, c_n\}$ and tripels $F = \{T_1, \dots, T_m\}$ ($|T_i| = 3$, $|T_i \cap A| = |T_i \cap B| = |T_i \cap C| = 1$). Decide, whether there is a *perfect 3-dim. matching*, i.e. a subset $F' \subseteq F$ of $|F'| = n$ disjoint tripels.

Let $t_j := |\{i \mid a_j \in T_i\}|$. We define an UNRELATED MACHINE SCHEDULING instance with machines $i = 1, \dots, m$ and the following set of jobs

- For $j = 1, \dots, n$ we have a job b_j with processing time

$$p_{i,b_j} = \begin{cases} 1 & \text{if } b_j \in T_i \\ \infty & \text{otherwise} \end{cases}$$

- For $j = 1, \dots, n$ we have a job c_j with processing time

$$p_{i,c_j} = \begin{cases} 1 & \text{if } c_j \in T_i \\ \infty & \text{otherwise} \end{cases}$$

- For every $j = 1, \dots, n$ we create jobs $D_{j,q}$ for $q = 1, \dots, t_j - 1$ with

$$p_{i,D_{j,q}} = \begin{cases} 2 & \text{if } a_j \in T_i \\ \infty & \text{otherwise} \end{cases}$$

Perform the following tasks

i) Show that if there is a perfect 3-dim matching, then the optimum makespan is at most 2.

ii) Show that if there is no perfect 3-dim maching, then the makespan is at least 3.

iii) Which inapproximability factor do you obtain for UNRELATED MACHINE SCHEDULING?

Solution:

i) Let F' be the perfect matching. If $T_i = (a_i, b_k, c_\ell) \in F'$ then we assign jobs b_k and job c_ℓ to machine i which then has a load of 2. Since F' covers every element in B and C exactly once, this assigns all B and C jobs. The $t_j - 1$ many jobs of the form $D_{j,q}$ are assigned to the $t_j - 1$ many free machines T_i with $a_j \in T_i$. These machines then also have a load of 2. Hence the makespan is 2.

ii) We show the contraposition which is

$$\text{makespan } \leq 2 \Rightarrow \exists \text{ 3-dim perfect matching}$$

Recall that the cumulated running times are $n+n+2\sum_{j=1}^{n}(t_j - 1) = 2\sum_{j=1}^{n} t_j = 2m$. We have m machines, hence each machine must be have a load of exactly 2. Thus every machine has either one job $D_{j,q}$ or one job b_k and one job c_ℓ. For any machine i, where the latter case happens we must have $b_k, c_\ell \in T_i$. Actually there is an a_j with $T_i = (a_j, b_k, c_\ell)$. Hence we choose F' as the set of T_i's, where this happens. The tripels in F' clearly cover every B and C element exactly once. So what happens to the A-elements. For any a_j we need $t_j - 1$ of the machines i with $a_j \in T_i$ to cover the jobs $D_{j,q}$. Hence at most one machine is used for the B and C jobs. In fact, exactly one machine must be used to handle B and C jobs. Hence every a_k will be contained in exactly one tripel.

iii) We obtain that for any $\varepsilon > 0$, there is no $(\frac{3}{2} - \varepsilon)$-approximation algorithm for UNRELATED MACHINE SCHEDULING.
