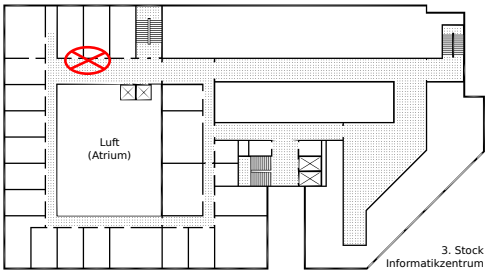
Algorithms Group **Summer '23**
Departement of Computer Science - IBR
TU Braunschweig

Prof. Dr. Sándor P. Fekete
Dr. Ahmad Moradi

Approximation Algorithms: Homework 1
10. May

Solutions are due on May, 24th until 11:00 AM in the cupboard for handing in practice sheets. Please put your name on all pages.



Exercise 1 (Factor Preserving Reduction):
Let Π_1 and Π_2 be two minimization problems. An *approximation factor preserving* reduction from Π_1 to Π_2 consists of two polynomial time algorithms, f and g, such that

- For any instance I_1 of Π_1, $I_2 = f(I_1)$ is an instance of Π_2 such that

$$OPT_{\Pi_2}(I_2) \leq OPT_{\Pi_1}(I_1)$$

- For any solution t of I_2, $s = g(I_1, t)$ is a solution of I_1 such that

$$Obj_{\Pi_1}(I_1, s) \leq Obj_{\Pi_2}(I_2, t).$$

Show that

a) Such reduction maps optimal solutions of I_2 to optimal solutions of I_1 and

$$OPT_{\Pi_2}(I_2) = OPT_{\Pi_1}(I_1)$$

b) If there is an α factor approximation algorithm for Π_2 then there is also an α factor approximation algorithm for Π_1.

(5+5 P.)

Exercise 2 (Expected Guarantee):
Let $\mathcal{A}$ be an algorithm for a minimization NP-optimization problem Π such that the expected cost of the solution produced by $\mathcal{A}$ is $\leq \alpha\ OPT$, for a constant $\alpha > 1$. What is the best approximation guarantee you can establish for Π using algorithm A? (*Hint*: First, apply Markov's inequality to bound the probability that the algorithm performs worse than expected by a factor of more than $(1 + \epsilon)$. Then, for guarantees arbitrarily close to α, run the algorithm polynomially many times and pick the best solution.)

(5+10 P.)

Exercise 3 (Set Cover):
Given a set of n elements, $\mathcal{E} = \{e_1, \ldots, e_n\}$, and a set of m subsets of $\mathcal{E}$, $\mathcal{S} = \{S_1, \ldots, S_m\}$ where $\cup_{i=1}^{m} S_i = \mathcal{E}$. A special case of the weighted set cover problem where cost of a cover is defined as the number of sets contained in the cover is called *cardinality* (or *unweighted*) set cover problem over the instance $(\mathcal{E}, \mathcal{S})$. A simple greedy idea for this special version of the problem reads

Algorithm 1: Greedy Cardinality Set Cover 1

Data: Sets $\mathcal{E}, \mathcal{S}$
Result: Set cover C
$C \leftarrow \emptyset$;
while $\mathcal{E}$ *containes elements not covered by* C **do**
  Pick a member $e \in \mathcal{E}$ not yet covered by C ;
  Add all sets S_i containing e to C ;
return C;

Prove that this is an F-approximation algorithm to the cardinality version of set cover defined above. Here, F, is the maximum *frequency* [1] of an element in $\mathcal{E}$ across $\mathcal{S}$. Could you provide an example showing your analysis is tight? Is this a constant factor aprroximation algorithm?

(10+10+5 P.)

Exercise 4 (Set Cover):
In the second lecture, we discussed a greedy idea to construct an approximation algorithm with factor $H_n = 1 + \frac{1}{2} + \cdots + \frac{1}{n}$ to the wighted set cover. Does the tight example we gave in the lecture work when all the sets are of cost equal to one? This motivates as well studying the same greedy idea for the cardinality set cover problem defined above. An adaptation of the greedy idea to the cardinality set cover reads

Algorithm 2: Greedy Cardinality Set Cover 2

Data: Sets $\mathcal{E}, \mathcal{S}$
Result: Set cover C
$C \leftarrow \emptyset$;
while $\mathcal{E}$ *containes elements not yet covered by* C **do**
  Find the set S_i containing the greatest number of uncovered elements ;
  Add S_i to C ;
return C;

Prove that It is again an approximation algorithm of logarithmically large factor and provide a tight example. How about the more specific case where all S_i are of size $\leq$ a given constant B. How do you compare the two approximation algorithms you see through exercise 3 and 4 for the cardinality set cover problem.

(10+10+5+5 P.)

[1] frequency of an element is simply the number of sets in $\mathcal{S}$ that contain that element.