Approximation Algorithms (236521) - Winter 2018
HW5 (Final)

Due on: 19.2.19 (Tuesday) by 18:00.

Guidelines:

• The assignment should be submitted individually.

• Discussing questions in this assignment with any individual (other than the course staff) is strictly prohibited.

• You may use claims shown in class. You may not use claims from previous home assignments.

• Claims from previous parts of a given question can be used even if they were not proved.

• You are required to provide complete and formal proofs of your claims.

• Please start each question in a new page.

• The assignment should be submitted electronically via the GR system.

Good Luck!

Question 1 (25 pt)

1. (10 pt) For an integer \( k \geq 2 \), a \( k \)-hypergraph is a pair \( H = (V, E) \), where \( V \) is a set of vertices and \( E \) is a set of hyperedges. A hyperedge \( e \in E \) is a non-empty subset of at most \( k \) vertices; that is, \( E \subseteq \{ S \subseteq V | S \neq \emptyset, |S| \leq k \} \).

A matching in a \( k \)-hypergraph \( H = (V, E) \) is a subset \( M \subseteq E \) such that for every \( e_1, e_2 \in M \), \( e_1 \neq e_2 \), it holds that \( e_1 \cap e_2 = \emptyset \).

A matching \( M \) is maximal if it is not strictly contained in another matching.

Give a polynomial time \( k \)-approximation algorithm for the problem of finding a minimum cardinality maximal matching in a \( k \)-hypergraph.

2. (15 pt) In the Facility Location problem we have \( m \) facilities and \( n \) clients. The cost of opening the \( i \)th facility is \( f(i) \geq 0 \), \( 1 \leq i \leq m \). If facility \( i \) is open, it can service client \( j \) at the cost \( c(i, j) \geq 0 \). The objective is to service all the clients at minimum total cost.

Formally, a solution is a subset \( S \subseteq \{1, \ldots, m\} \) of facilities and a mapping \( \mu : \{1, 2, \ldots, n\} \rightarrow S \) of clients to facilities in \( S \). The cost of a solution is \( \sum_{i \in S} f(i) + \sum_{j=1}^{n} c(\mu(j), j) \).

(a) (4 pt) Show that when \( c(i, j) \in \{0, \infty\} \) for all \( 1 \leq i \leq m \) and \( 1 \leq j \leq n \) the problem is equivalent to Set Cover. Formally, in this case, any instance \( I \) of Facility Location can be transformed to an instance \( I' \) of Set Cover, such that a solution for \( I \) of finite cost induces a feasible solution for \( I' \) of the same cost and vice versa. You may assume that the instance \( I \) always has a solution of finite cost.

Use the following definition for Set Cover. The input is a universe \( U \) along with \( m \) subsets \( S = \{S_1, S_2, \ldots, S_m\} \), where \( S_i \subseteq U \) for \( i = 1, 2, \ldots, m \). Each subset \( S_i \) is associated with a cost \( c_i \geq 0 \). A feasible solution for Set Cover is a subset \( T \subseteq S \) such that \( \cup_{S_i \in T} S_i = U \). The cost of the solution \( T \) is \( \sum_{S_i \in T} c_i \).
Consider the following algorithm for Facility Location:

Find a facility \( i \) and a subset of clients \( S \) that minimize the ratio \( \frac{f(i) + \sum_{j \in S} c(i, j)}{|S|} \).

Open facility \( i \) and use it to service the clients in \( S \). Omit the subset \( S \) of clients from the input (but not facility \( i \)). Continue similarly until all clients are serviced.

(b) (4 pt) Show that the algorithm achieves a ratio of \( O(\log n) \) to the minimum cost.

(c) (7 pt) Show that the algorithm can be implemented in polynomial time.

**Question 2 (25 pt)**

In this question we consider the problem of Bin Packing in Pairs. It is a variant of the Bin Packing problem taught in class with the following modification. There are bins of capacities \( B_1 \) and \( B_2 \). The objective is to pack the items in a minimal number of pairs of bins, where each pair consists of one bin of capacity \( B_1 \) and one bin of capacity \( B_2 \).

Formally, the input consists of a set of \( n \) items \( I = \{1, 2, \ldots, n\} \), each item \( i \in I \) is associated with a size \( s_i > 0 \). The input includes also two bin capacities \( B_1 \geq B_2 > 0 \), such that \( B_1 \geq s_i \) for all \( i \in I \). A solution is a partition of \( I \) into 2\( m \) sets \( U_1, \ldots, U_m \) and \( V_1, \ldots, V_m \) such that for any \( 1 \leq j \leq m \) it holds that \( \sum_{i \in U_j} s_i \leq B_1 \) and \( \sum_{i \in V_j} s_i \leq B_2 \). The cost of the solution is \( m \), the number of pairs of bins used for the packing.

1. (6 pt) Find constants \( c_1, c_2 \geq 0 \), and an algorithm \( A_\varepsilon \) for any \( 0 < \varepsilon < 1/2 \), such that
   - \( A_\varepsilon \) is polynomial for any fixed \( \varepsilon \).
   - For any input of Bin Packing in Pairs in which \( B_2 \leq \varepsilon B_1 \), if the optimal solution has the cost \( m \) then \( A_\varepsilon \) returns a solution of cost at most \( m(1 + c_1 \varepsilon) + c_2 \).

2. (16 pt) Find constants \( d_1, d_2 \geq 0 \), and an algorithm \( A_\varepsilon \) for any \( 0 < \varepsilon < 1/2 \), such that
   - \( A_\varepsilon \) is polynomial for any fixed \( \varepsilon \).
   - For any input of Bin Packing in Pairs in which \( B_2 > \varepsilon B_1 \), if the optimal solution has the cost \( m \) then \( A_\varepsilon \) returns a solution of cost at most \( m(1 + d_1 \varepsilon) + d_2 \).

**Hint:** Try to divide the items into large and small, as in the APTAS for Bin Packing.

3. (3 pt) Give an APTAS for Bin Packing in Pairs.

**Question 3 (25 pt)**

In the Multidimensional Knapsack problem we are given a \( d \)-dimensional knapsack, for some \( d \geq 2 \), having capacity \( b_i \) in the \( i \)th dimension, for \( 1 \leq i \leq d \). Also, there is a set of items \( I = \{1, \ldots, n\} \). The size of item \( j \) in dimension \( i \) is \( s_{ij} > 0 \), \( 1 \leq j \leq n \), \( 1 \leq i \leq d \). The profit from packing item \( j \) is \( p_j > 0 \). A solution is a subset of items \( A \subseteq I \), and the profit of \( A \) is \( P(A) = \sum_{j \in A} p_j \). A solution \( A \) is feasible if \( \sum_{i \in A} s_{ij} \leq b_i \) for all \( 1 \leq i \leq d \). We assume below that \( d \) is a fixed constant. Consider the following algorithm for \( d \)-dimensional Knapsack.

Let \( \varepsilon > 0 \) and \( h = \min\{n, \lceil d(1 + \varepsilon)/\varepsilon \rceil\} \). Given a selected subset of items \( S \subseteq I \), the objective of the \( S \)-problem is to find a feasible packing of items in the knapsack which maximizes the profit, subject to the following constraints:

(i) All items in \( S \) are packed.
(ii) Among the items in \( I \setminus S \) only items \( j \) with profit \( p_j \leq \min_{\ell \in S} p_\ell \) can be added to the solution. Denote this set of items by \( \text{Good}(S) \).

Given the selected subset of items \( S \), consider \( LP(S) \) to be a linear programming relaxation of the \( S \)-problem. Let \( x_j \) be an indicator for the selection of item \( j \in I \) for the solution. In the linear programming relaxation \( LP(S) \), \( 0 \leq x_j \leq 1 \). Let \( x^B = (x_1^B, \ldots, x_n^B) \) be a basic solution for \( LP(S) \).

The Algorithm: For any \( S \subseteq I \) such that \( |S| \leq h \), solve optimally \( LP(S) \) and round down \( x^B \).

Choose the solution which maximizes the profit.

1. (6 pt) Formulate the linear program \( LP(S) \).
2. Denote by \( x^I_j \) the integer variable obtained after rounding the basic variable \( x^B_j \). Let \( F \) be the subset of items \( j \) for which \( x^B_j \) is fractional, i.e., \( x^I_j < x^B_j \).
   (a) (8 pt) Show that \( |F| \leq d \).
   (b) (2 pt) Show that if \( |S| = h \) then for all \( j \in F \) it holds that \( p_j \leq \frac{1}{h} \sum_{\ell \in S} p_\ell \).
3. Let \( OPT \) be a subset of items packed in an optimal solution. In this part you will show that the algorithm achieves a ratio of \( \frac{1}{1 + \varepsilon} \) to the optimal profit, by proving the next claims.
   (a) (2 pt) If \( |OPT| \leq h \) then the algorithm outputs an optimal solution.
   (b) (3 pt) If \( |OPT| > h \), there exists a subset of \( h \) items \( S^*_h \), \( |S^*_h| \leq h \), such that the rounded solution for \( LP(S^*_h) \) satisfies

\[
P(OPT) \leq \sum_{j=1}^{n} p_j x^I_j + \delta,
\]

where \( \delta = \frac{d}{h} \sum_{\ell \in S^*_h} p_\ell \).

(c) (4 pt) Show that this implies the approximation ratio of \( \frac{1}{1 + \varepsilon} \) for the algorithm.

Question 4 (25 pt)

Let \( G = (V, E) \) be a connected undirected graph and \( T_1, T_2, \ldots, T_t \subseteq V \). Let \( w : E \rightarrow \mathbb{R}_+ \) be a non-negative weight function over the edges. A generalized Steiner forest is a subset \( E' \subseteq E \), such that for every \( u, v \in T_i \) (1 \( \leq i \leq t \)) there is a path in \( E' \) connecting \( u \) and \( v \). In the GENERALIZED STEINER FOREST problem the objective is to find a generalized Steiner forest of minimum weight.

1. (12 pt) Let \( T = \bigcup_{i=1}^{t} T_i \). Assume \( |T_i| \geq 2 \) for any \( 1 \leq i \leq t \) and consider \( w' : E \rightarrow \mathbb{R} \) defined as follows. For an edge \( (u, v) \), \( w'(u, v) = |\{u, v\} \cap T| \). Show that any minimal generalized Steiner forest of \( G \) is a 2-approximation with respect to the weight function \( w' \).

   Note: a generalized Steiner forest \( E' \) of \( G \) is minimal if there is no \( F \subset E' \) which is also a generalized Steiner forest of \( G \).

2. (13 pt) Suggest a 2-approximation algorithm for the Generalized Steiner Forest problem using the Local Ratio technique.
You may use the following construction in your answer.

Given a graph $G = (V, E)$, a weight function $w : E \to \mathbb{R}_+$ and an edge $(u, v) \in E$ we define the contraction of $G, w$ over the edge $(u, v)$ as the graph $G' = (V', E')$ with a weight function $w' : E' \to \mathbb{R}_+$ along with three functions $f : V \to V'$, $\phi : E' \to E$ and $\phi' : E \setminus \{(u, v)\} \to E'$, defined as follows:

- $V' = V \setminus \{u, v\} \cup \{v^*\}$, where $v^* \notin V$ is a new vertex.
- $f(z) = z$ for $z \in V \setminus \{u, v\}$ and $f(u) = f(v) = v^*$.
- $E' = \{(f(x), f(y)) | (x, y) \in E, f(x) \neq f(y)\}$.
- $\phi'(x, y) = (f(x), f(y))$ for any $(x, y) \in E \setminus \{(u, v)\}$.
- $\phi(e') = \text{argmin}_{e | \phi'(e) = e'} w(e)$ for any $e' \in E$. If the minimum is attained with multiple edges pick one arbitrarily.
- $w'(e') = w(\phi(e'))$ for any $e' \in E'$.

Define $\forall F \subseteq E : w(F) = \sum_{e \in F} w(e), \forall F' \subseteq E' : w'(F') = \sum_{e' \in F'} w'(e'), \forall F \subseteq E \setminus \{(u, v)\} : \phi'(F) = \{\phi'(e) | e \in F\}, \forall F' \subseteq E' : \phi(F') = \{\phi(e') | e' \in F\}$.

The following properties hold with respect to the above contraction:

(P1) For any $F' \subseteq E'$, it holds that $w(\phi(F')) \leq w'(F')$.

(P2) For any $F \subseteq E \setminus \{(u, v)\}$, it holds that $w'(\phi'(F)) \leq w(F)$.

(P3) Let $F' \subseteq E'$. For any $x, y \in V$, if there is a path in $F'$ connecting $f(x)$ and $f(y)$ then there is a path in $\phi(F') \cup \{(u, v)\}$ connecting $x$ and $y$.

(P4) Let $F \subseteq E \setminus \{(u, v)\}$. For any $x, y \in V$, if there is a path in $F \cup \{(u, v)\}$ connecting $x$ and $y$ then there is a path in $\phi'(F)$ connecting $f(x)$ and $f(y)$.