Australian National University
Research School of Computer Science

# COMP4600 in 2014 – Assignment three

**Due:** 12:00 (at noon), Friday, Oct. 27
**Late Penalty:** 25% per day

*Put your work as an attachment of an email to wliang@cs.anu.edu.au with the subject title COMP4600 Assignment 3.* The total marks for this assignment are 50 points, which is worth 18% of the final mark.

**Question 1** (10 points)

Solve the following linear program using SIMPLEX:

$$
\begin{array}{ll}
\text{minimize} & x_1 + x_2 + x_3 \\
\text{subject to} & \\
& 2x_1 + 7.5x_2 + 3x_3 \geq 10000 \\
& 20x_1 + 5x_2 + 10x_3 \geq 30000 \\
& x_1, x_2, x_3 \geq 0.
\end{array}
$$

**Question 2** (7 points)

Write the dual to the following linear program.

$$
\begin{array}{ll}
\text{maximize} & x + y \\
& 2x + y \leq 3 \\
& x + 3y \leq 5 \\
& x, y \geq 0.
\end{array}
$$

Find the optimal solutions to both primal and dual LPs.

**Question 3** (7 points)
Consider the following generalization of the maximum flow problem.

You are given a directed network $G = (V, E)$ with edge capacities $c_e$ for each edge $e \in E$. Instead of a single $(s, t)$ pair, you are given multiple pairs $(s_i, t_i)$, where node $s_i$ is the source and node $t_i$ is the destination (sink) of source $s_i$, and each source $s_i$ has a demand $d_i > 0$ to be routed to its destination $t_i$, $1 \leq i \leq k$. The goal is to find $k$ flows $f^{(1)}, \ldots, f^{(k)}$ with the following properties.

- $f^{(i)}$ is a valid flow from $s_i$ to $t_i$.
- For each edge $e$, the total flow $\sum_{i=1}^{k} f_e^{(i)} \leq c_e$.
- The size of each flow $f^{(i)}$ is at most the demand $d_i$.
- The size of the total flow (the sum of the flows) is as large as possible.

How would you solve this problem?

**Question 4** (7 points)

The weighted set-covering problem is defined as follows. Given a set $S = \{a_1, a_2, \ldots, a_n\}$ containing $n$ elements and there is a family $\mathcal{F}$ of subsets of $S$, i.e., $\mathcal{F} = \{S_i \mid 1 \leq i \leq m\}$ where $S_i \subset S$ and $\cup_{i=1}^{m} = S$, associated with each subset $S_i \in \mathcal{F}$, there is weight $w_i > 0$, the weight of a cover $\mathcal{C}$ is $\sum_{S_i \in \mathcal{C}} w_i$ where $\mathcal{C} \subseteq \mathcal{F}$, the problem is to find a minimum-weight cover. Note that the set-covering problem is a special case of this general setting when $w_i = 1$ for all $S_i \in \mathcal{F}$.

**(i)** How to generalize the greedy set-covering heuristic in a natural manner to provide an approximate solution for any instance of the weighted set-covering problem.

**(ii)** Show the approximation ratio of the algorithm is $H(d)$, where $d$ is the maximum size of any set $S_i$.

**Question 5** (7 points)

Given a set of $n$ points $V = \{v_1, v_2, \ldots, v_n\}$ in a 2 dimensional plane, the weight between two points $v_i$ and $v_j$ is the Euclidean distance between them for all $i$ and $j$ with $1 \leq i, j \leq n$. Let OPT be the weight sum of of the edges in a minimum spanning tree (MST) $T$ of the complete graph $K[V]$ induced by the points in $V$. Let $U$ be a proper subset of $V$ ($U \subset V$) and $C(U)$ the weight sum of of the edges in a MST of the complete subgraph $K[U]$ induced by the nodes in $U$. Show the cost $C(U)$ of the MST of $K[U]$ is at most twice the cost of the MST of $K[V]$, i.e.,

$$C(U) \leq 2 \cdot \text{OPT}.$$

**Question 6** (7 points)

Given a set of $m$ machines $M_1, M_2, \ldots, M_m$ and a set of $n$ jobs, each job $j$ has a processing time $t_j$, $1 \leq j \leq n$. We seek to assign each job to one of the $m$ machines so that the work loads placed on all machines are as "balanced" as possible.

Let $A(i)$ denote the set of jobs assigned to machine $M_i$. Under this assignment, machine $M_i$ needs to work for a total time of $T_i = \sum_{j \in A(i)} t_j$, which is the load at machine $M_i$. We seek to minimize a quantity known as *the makespan*, i.e., the maximum load among all machines $T = \max_i \{T_i \mid 1 \leq i \leq m\}$ is minimized.

The following sort-based heuristic can provide an approximate solution to the problem.

**Sort_Balance**($t_j$)

**for** $i = 1$ to $m$ **do**<br>
    $T_i \leftarrow 0$;<br>
    $A(i) \leftarrow \emptyset$;<br>
**endfor**;<br>
Sort jobs in decreasing order of processing time $t_j$, assuming $t_1 \geq t_2 \geq \ldots \geq t_n$;<br>
**for** $j = 1$ to $n$ **do**<br>
    Let $M_i$ be a machine that achieves the minimum $\min_k T_k$;<br>
    Assign job $j$ to machine $M_i$;<br>
    $A(i) \leftarrow A(i) \cup \{j\}$;<br>
    $T_i \leftarrow T_i + t_j$;<br>
**endfor**

Show the approximation ratio of algorithm `Sort_Balance` is 3/2. *Hint: If there are more than $m$ jobs, then $T^* \geq 2t_{m+1}$, where $T^*$ is the optimal makespan.*

**Question 7** Choose **one** of the following two questions to work (5 points).

**Q8 (a)** Given an undirected bipartite graph $G(V, E)$, show that the cardinality of a maximal matching of $G$ is no less than half the cardinality of the maximum matching of $G$. A maximal matching of $G$ is such a matching that any further addition of edges to it will violate the edge matching constraint.

**Q8 (b)** Given an undirected, connected graph $G = (V, E)$, partition the nodes in $V$ into two disjoint sets $S_1$ and $S_2$ such that the number of edges between the nodes in $S_1$ and $S_2$ is maximized. In the following, there is a simple greedy algorithm of finding such a partition for this problem.

**Local_Search_MAX-CUT**($G(V, E)$)

$(S_1, S_2) \leftarrow$ any partition of the set $V$;
/* Let $d_{in}(v)$ and $d_{out}(v)$ be the degrees
    of node $v$ in its own partition and another partition;
**while** $\exists u \in V$ s.t. $d_{in}(u) > d_{out}(u)$ **do**
    **if** $u \in S_1$ **then** move node $u$ to $S_2$;
    **else**
        move node $u$ to $S_1$
    **endif**;
**endwhile**
**Return** $(S_1, S_2)$.

Show that the solution obtained $(S_1, S_2)$ by algorithm `Local_Search_MAX-CUT` is at least 1/2 of the optimal.