上 海 交 通 大 学 试 卷（A卷）

（ 2016至 2017学年 第2学期 ）

学　　号：　　姓名：

课程名称：　算法设计与分析　　成绩：

1. (10 points) Write the dual to the following linear program.

$$\text{Max } x/2 + 2y - 4$$
$$x + y \leq 3$$
$$x/2 + 3y \leq 5$$
$$x, y \geq 0$$

Find the optimal solutions to both primal and dual LPs.

Sol:

The dual program:

Min 3x+5y-4

subject to x+1/2y ≥ 1/2

x+3y ≥ 2

x, y ≥ 0

the optimal solution of primal: (8/5,7/5) (max = -2/5)

the optimal solution of dual: (1/5,3/5)(min=-2/5)

我承诺，我将严格遵守考试纪律。

承诺人：_____

题号									
得分									
批阅人(流水阅卷教师签名处)									

2. (15 points) Find a polynomial time 4/3-approximation algorithm for instance of **Metric TSP** where the distances are either 1 or 2.
   Hint: The 2-match problem (find a minimum weight subset of edges M such that each node is adjacent to exactly 2 edges in M) can be solved in polynomial time.
   Sol:
   Algorithm:
   1. Find the minimum 2-matching M
   2. If M is connected then return M else then delete one edge in each component and connect the component then return.

   Algorithm analysis:
   Let minM be the length of M and M has k component. The answer of the algorithm is ANS and the OPT be the optimal tour length. We just need to prove that $\frac{4}{3}OPT \geq ANS$.

   First, since any TSP tour is a 2-matching we have $ANS \leq OPT$. Therefore, if M is connected then it is an optimal TSP tour. Otherwise, M has k(>1) components. Notice that each components has at least 3 nodes, so $k \leq \frac{n}{3}$. This means that when we change the edge from M, the increase cut is at most k since the distances are either 1 or 2. So we have:

$$ANS \leq OPT + k \leq OPT + \frac{n}{3} \leq OPT + \frac{1}{3}OPT \leq \frac{4}{3}OPT \ \#$$

3. (15 points) Consider the following algorithm for **Cardinality Vertex Cover**: In each connect component of the input graph execute a **depth first search (DFS)**. Output the nodes that are not leaves in the DFS tree. Show that the output is indeed vertex cover, and that it approximates the minimum vertex cover within a factor of 2.

Sol: Let T be the DFS tree, L be the set of leaves, n be the total number of vertices. First we prove that V-L is a vertex cover. Since all edges are either forward edges or back edges, there is no edges between leaves, hence V-L is a valid vertex cover.

To see it approximates the minimal vertex cover within a factor of 2, we need to prove that for any vertex cover VC, we have：

$$|VC| \geq \frac{n-|L|}{2}$$

For the dfs tree we can assume that none of the leaves are in the optimal vertex cover since it's always less beneficial. Notice that each edge in the tree should be matched in VC, then for any vertex cover, they must contain at least half about the nodes that are note leaves in the dfs tree.

4. (15 points) Prove that the **Graph-Isomorphism problem** is a NP problem.

Sol:

GI: *Given two graphs G1 and G2 and ask whether G1 is isomorphic to G2*

GI in NP is very simple. The certificate is $\phi : V_1 \to V_2$ .The verifiying algorithm checks if ϕ is a bijection and for all $u, v \in V_1$ whether $(u, v) \in E_1$ if and only if $(\phi(u), \phi(v)) \in E_2$

5. (15 points) **Set Multicover problem** is defined as an extension of **Set Cover problem**, such that each element, e, needs to be covered a specified integer number, r_e, of times. The objective again is to cover all elements up to their coverage requirements at minimum cost. We will assume that the cost of picking a set S k times is $k \cdot cost(S)$. Represent the problem as an integer linear program. Then relax to a LP and work out its dual.

Sol: The integer linear program:

$$\begin{aligned} & \min \sum_{S_i \in S} cost(S) \cdot x_{s_i} \\ \textit{subject to} \quad & \forall e \in U \quad \sum_{S:a \in S} x_S \geq r_e \\ & \forall S_i \in S \quad x_s \in N \end{aligned}$$

Relax to a LP:

$$\begin{aligned} & \min \sum_{S_i \in S} cost(S) \cdot x_{s_i} \\ \textit{subject to} \quad & \forall e \in U \quad \sum_{S:a \in S} x_S \geq r_e \\ & \forall S_i \in S \quad x_s \geq 0 \end{aligned}$$

The LP's dual:

$$\begin{aligned} & \max \sum_{e \in U} y_e \cdot r_e \\ \textit{subject to} \quad & \forall S_i \in S \quad \sum_{a \in S_i} x_S \leq cost(S) \\ & \forall e \in U \quad y_e \geq 0 \end{aligned}$$

6. (15 points) Given two strings $x = x_1x_2 \ldots x_n$ and $y = y_1y_2 \ldots y_m$, we wish to find the length of their longest common substring, that is, the largest k for which there are indices i and j with $x_ix_{i+1} \ldots x_{i+k-1} = y_jy_{j+1} \ldots y_{j+k-1}$. Show how to do this in time $O(mn)$.

Sol: Let L[i][j] be the maximal length of the common subsequence of $x_1x_2 \ldots x_i$ *and* $y_1y_2 \ldots y_j$ Then L[i][j] satisfies:

$$L[i][j] = \begin{cases} 0 & if\ i = 0\ or\ j = 0 \\ L[i-1][j-1] + 1 & if\ \text{a[i]=b[j]} \\ 0 & \text{o.w.} \end{cases}$$

Then we can get the algorithm:

For i=1 to n

Update L[i][0]

For i=1 to m

Update L[0][i]

For i=1 to n

For j=1 to m

Update L[i][j]

Return max L[i][j]

It can be easily seen that the algorithm can be finished in $O(nm)$.

7. (15 points) Suppose in a given network, all edges are undirected (or think of every edge as bi-directional with the same capacity), and the length of the longest simple path from the source s to sink t is at most L. Show that the running time of **Edmond-Karp algorithm** on such networks can be improved to be $O(L \cdot |E|^2)$.

Sol:

Algorithm 2 EDMONDSKARP(G)

1: $f \leftarrow 0$; $G_f \leftarrow G$
2: **while** G_f contains an $s-t$ path P **do**
3: Let P be an $s-t$ path in G_f with the minimum number of edges.
4: Augment f using P.
5: Update G_f
6: **end while**
7: **return** f

The key ideas are:

- In each iteration in the body of while loop (line 3-5) the running time is O(E)
- The iteration times is at most O(LE)
  - In each iteration, at least one edge on the augmenting path must be *critical* (An edge is critical if the residual capacity of the augmenting path p equals the residual capacity of the edge (u, v), that is, $c_f(p) = c_f(u, v)$).
  - From the time (u, v) becomes critical to the time when it next becomes critical, the distance of u from the source increases by at least 2. Since the distance of u from the source is at most L, an edge can become critical at most O(L) times. (Need elaboration with mathematical proofs)
  - There are O(E) pair of vertices that can have an edge between them in a residual network
  - Each augmenting path has at least one critical edge

So, the total time is $O(L \cdot |E|^2)$.

For a more detailed proof, see the book **Introduction to Algorithms**, from page 727 to page 730. http://basics.sjtu.edu.cn/~liguoqiang/teaching/Galgo17/books/IntroductiontoAlgorithms3rd.pdf