Name(last, first): _____

U C L A Computer Science Department

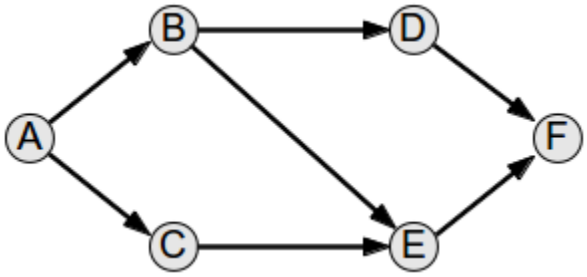
CS 180 **Algorithms & Complexity** **ID (rightmost 4 digits):**

Final Exam **Total Time: 3 hours** **December 6, 2016**

1. (20 points).

a. (10 pts) Consider an S-T network N. Prove that if there is no augmenting path in the residual graph G*f* (which by definition has *f* units of flow), then there is a S-T cut with its capacity equal to flow *f*.

Problem 1 b. (10 pts) If the max flow algorithm initially finds the path A,B,E,F in the graph below, show the residual graph and all subsequent steps of Ford-Fulkerson algorithm on this graph (all capacities are 1). .



2. (15 points).

N teams attend a dinner. Team **i** has $\mathbf{t_i}$ members. There are **M** tables at the dinner, with $\mathbf{M \geq N}$. Table **i** has $\mathbf{s_i}$ chairs. We wish to seat all teams such that no two team-members are at the same table. **a. (12 pts)** Design a polynomial-time algorithm that solves this problem (Hint: Use network flow). **b. (3 pts)** Provide time complexity analysis.

3. (15 points). You are given a sequence of integers **X** = (x1, x2, …, xn) and a total sum **S**. Design an algorithm that decides if there is a subset of these numbers that add up to S. Show your algorithm step-by-step (in bullet format).

.

Name(last, first): _____

4. (15 points). Consider a set of line segments bounded by two horizontal lines. Design an algorithm that finds the maximum number of non-crossing line segments. (In the above example the answer contains 6 segments).

B Analyze the time complexity of your algorithm.

5. (20 points).

a. (2 pts) Does Maximum Clique require exponential time to solve? Explain.

b. **(8 pts)** If Y is polynomial time transformable to X and Y cannot be solve in polynomial time, what can we state about X? **Prove** your answer.

Problem 5 c. (10 pts) Prove that **Vertex cover** is polynomial time transformable to **set cover**.

Name(last, first): _____

6. (15 points). Given a sorted array X = (x1, x2, …, xn) in which all elements appear twice (one after the other) and one element appears only once. Design an algorithm that finds the number that appears only once. Note: a linear-time algorithm would be trivial. Find a more efficient algorithm. Describe your algorithm bullet-by-bullet.
Example:

```
Input:   arr[] = {1, 1, 3, 3, 4, 5, 5, 7, 7, 8, 8}
Output:  4
```