Final Exam

Instructions:

Please write clearly and succinctly; be sure to give an overview before plunging into details

Your name:

On problem	you got	out of
1		20
2		20
3		25
4		20
5		15
Σ		100

1 Scheduling [20 points]

Suppose that you have n unit length jobs to schedule on m processors. Typically $n >> m$. The *ith* job has a list L_i of the processors on which it can be run. Each job i is to be assigned to exactly one processor on its list L_i.

Multiple jobs can be assigned to the same processor, however, each processor j has a job limit P_j which is the maximum number of jobs that can be assigned to processor j.

(a) Describe an efficient algorithm which either finds a legal assignment of all jobs or determines that no such assignment exists. Be sure to justify the correctness of your solution algorithm (Hint: greedy and dynamic programming solutions are unlikely to be correct).

(b) Give the running time of your solution as a function of n and m.

2 Suffix Trees [20 points]

In our description of suffix trees we claimed that once a suffix tree is built, we can find all positions where a pattern P of length m occurs in $O(m+k)$ time, where k is the number of times the Pattern P occurs. This claim is not immediate since you will normally find the end of P in the middle of the tree and will then have to go down to the leaves to find where it occurs.

A) Show how to find all occurances of the pattern in $O(m+k)$ time. Justify the correctness and time bound of your solution (note, no modifications to the suffix tree should be necessary).

B) If we only want to find the last occurance of P how should the suffix tree be modified so that we can find the position of the last match to P in $O(1)$ time once we have found P in our tree?

3 Network Flow [25 points]

Part A. Suppose that all arcs in our flow network have capacities of one. We call this a *Unit flow network.* The preflow push will now have much better worst case performance. Prove a sharper bound on the performance of the Generic preflow-push algorithm for a unit network with n nodes and m arcs.

Part B. We define a *bipartite flow network* as follows: its vertices are s, t and two sets of vertices L and R. All arcs in the network go from s to nodes in L, from a node in L to a node in R, or from a node in R to t.

i) Let l be the number of nodes in L and r be the number of nodes in R. Suppose that $l >> r$. In a general flow network an Augmenting path can be as long as $n - 1$ (where for this network $n = r + l + 2$). Give a much better bound for the maximum length of an augmenting path in a Bipartite flow network. Justify your answer.

ii) If we have a flow network N where the maximum length of any simple path in the residual network is d, what can we say about the maximum height of a node in N during the course of the preflow push algorithm?

4 NP-Completeness Bound [20 points]

The *3-Disk Packing (3DP)* problem is as follows: you want to store n files on three equal size disks where each file is to reside entirely on one disk. The input is n, the file sizes $f_1, f_2, \ldots, f_n$ in bytes, and D the capacity of each of the three disks (also in bytes). The output is yes/no depending on whether such a packing exists.

A) Show that 3DP is in NP.

B) Use the fact that Set Partition is NP-complete to show that 3DP is also NP-complete.

C) If you want to show that a problem Q is NP-hard would it ever be better to use Subset Sum instead of Set Partition? Justify your answer.

5 Approximatation algorithms [15 points]

We consider variants on the approximation algorithms we discussed for the Traveling Salesman Problem (TSP). The *Repetition Traveling Salesman Problem (RTSP)* is the same as the TSP problem (we have a complete weighted graph as input, and want the shortest cycle which visits each city at least once), but we are now allowed to visit cities more than once on the tour if that is helpful.

A) Suppose that we have a complete graph G with non-negative symmetric costs but which does NOT obey the triangle inequality. Describe a variant of the 1-approximation algorithm for TSP we described in class which is a 1-approximation algorithm for RTSP in this setting.

B) Justify the correctness of your approximation algorithm of part A.