

# CSE548/AMS542 Spring 2008 Analysis of Algorithms

Jie Gao\*

April 29, 2008

Due **May 13th**. Each problem, unless specified otherwise, has a maximum of 10 points. (i) You write down the solution clearly. If we can not recognize your writing then you may lose points. (ii) Avoid too many details. A succinct and clean proof is the best. You may use the algorithms we covered in class without referring to the details. (iii) If you discuss some of the problems with another fellow student (at most 2 students per group), write down his/her name and the problems. If you consult any books/webpages, cite them.

## Homework 6

1. Consider the load balancing problem in which jobs are assigned to machines. Suppose now that there are two types of machines, the *fast* machines run twice faster than the *slow* machines. You have  $k$  fast machines and  $m$  slow machines and you have  $n$  jobs with job  $i$  taking time  $t_i$  on a slow machine. The problem is to assign the jobs to the machines so that all jobs can be finished as soon as possible.
  - (a) Prove that this problem is NP-hard. (5pts)
  - (b) Given a polynomial 3-approximation algorithm. (10pts)
2. Given a set  $A = \{a_1, a_2, \dots, a_n\}$  and sets  $B_1, B_2, \dots, B_m$  as subsets of  $A$ , we say a set  $H \subseteq A$  is a *hitting set* if  $H \cap B_i \neq \emptyset$  for any  $i$ .
  - (a) Prove that finding the hitting set with minimum cardinality is NP-complete. (5pts)
  - (b) Now suppose each element  $a_i$  has a weight  $w_i \geq 0$ . Now we would like to find a hitting set with the total weight as small as possible. Show a polynomial algorithm that finds a hitting set with total weight at most  $b$  times the optimal, where  $b = \max_i |B_i|$ . (10pts)
3. Consider the 3D matching problem. Given three disjoint sets  $X, Y, Z$  and a set  $T \subseteq X \times Y \times Z$  as a subset of ordered triples. A subset  $M \subseteq T$  is a 3D matching if each element of  $X, Y, Z$  is contained in at most one of these triples. We would like to find the matching of maximum size. Give a polynomial algorithm that finds a matching of size at least  $1/3$  that of the optimal solution.
4. Consider a graph  $G = (V, E)$  with  $|V|$  as multiples of 3, a 3-coloring scheme assigns a color out of three possible colors to each vertex such that no two adjacent vertices have the same color.
  - (a) Now we also require that there are exactly  $|V|/3$  vertices of each color and call the coloring scheme a balanced coloring scheme. Prove that testing whether a graph has a balanced 3-coloring scheme is NP-complete. (10pts)

---

\*Department of Computer Science, Stony Brook University, Stony Brook, NY 11794, USA, [jgao@cs.sunysb.edu](mailto:jgao@cs.sunysb.edu).

- (b) Give a randomized algorithm that colors the vertices such that there are at least  $2/3$  edges (in expectation) with differently colored vertices. (10pts)
5. Prove that the  $k$ -center problem on a weighted graph is NP-complete. That is, given a graph  $G = (V, E)$  with integer edge weights, find a subset of  $k$  vertices (call them centers) such that any vertex is within distance  $d$  from at least one center. The  $k$ -center problem asks whether such a collection of centers can be found or not.
6. Consider a random walk on a path with vertices numbered  $1, 2, \dots, n$  from left to right. At each step, we flip a coin to decide which direction to walk, moving one step left or one step right with equal probability. The random walk ends when we fall off one end of the path, either by moving left from vertex 1 or by moving right from vertex  $n$ .
- (a) Prove that if we start at vertex 1, the probability that the walk ends by falling off the left end of the path is exactly  $n/(n+1)$ . (5pts)
- (b) Prove that if we start at vertex 1, the expected number of steps before the random walk ends is exactly  $n$ . (5pts)
- (c) Suppose we start at vertex  $n/2$  instead. State a tight bound on the expected length of the random walk in this case. (5pts)
7. (Extra credit) Suppose you have a blackbox that can tell in constant time whether a graph has a Hamiltonian cycle or not. Use this blackbox as a subroutine and show that you can actually find a polynomial algorithm that *computes* a Hamiltonian cycle of a graph.
8. (Extra credit) Let  $X[1..n]$  be an array of  $n$  distinct real numbers, and let  $N[1..n]$  be an array of indices with the following property: If  $X[i]$  is the largest element of  $X$ , then  $X[N[i]]$  is the smallest element of  $X$ ; otherwise,  $X[N[i]]$  is the smallest element of  $X$  that is larger than  $X[i]$ . Describe and analyze a randomized algorithm that determines whether a given number  $x$  appears in the array  $X$  in  $O(\sqrt{n})$  expected time.