CS 473U: Undergraduate Algorithms, Fall 2006
Homework 6

Due at **4 p.m. on** Friday, November 17, 2006 in 3229 Siebel Center

---

Remember to turn in separate, individually stapled solutions to each of the problems.

---

1. Given an undirected graph $G(V, E)$, with three vertices $u, v, w \in V$, you want to know whether there exists a path from u to w via v. (That is, the path from u to w must use v as an intermediate vertex.) Describe an efficient algorithm to solve this problem.

2. *Ad-hoc Networks*, made up of cheap, low-powered wireless devices, are often used on battlefields, in regions that have recently suffered from natural disasters, and in other situtations where people might want to monitor conditions in hard-to-reach areas. The idea is that a large collection of the wireless devices could be dropped into the area from an airplane (for instance), and then they could be configured into an efficiently functioning network.

   Since the devices are cheap and low-powered, they frequently fail, and we would like our networks to be reliable. If a device detects that it is likely to fail, it should transmit the information it has to some other device (called a *backup*) within range of it. The range is limited; we assume that there is a distance d such that two devices can communicate if and only if they are within distance d of each other. To improve reliability, we don't want a device to transmit information to a neighbor that has already failed, and so we require each device v to have at least k backup devices that it could potentially contact, all of which must be within d meters of it. We call this the *backup set* of v. Also, we do not want any device to be in the backup set of too many other devices; if it were, and it failed, a large fraction of our network would be affected.

   The input to our problem is a collection of n devices, and for each pair u, v of devices, the distance between u and v. We are also given the distance d that determines the range of a device, and parameters b and k. Describe an algorithm that determines if, for each device, we can find a backup set of size k, while also requiring that no device appears in the backup set of more than b other devices.

3. **UPDATED:** Given a piece of text T and a pattern P (the 'search string'), an algorithm for the string-matching problem either finds the first occurrence of P in T, or reports that there is none. Modify the Knuth-Morris-Pratt (KMP) algorithm so that it solves the string-matching problem, even if the pattern contains the wildcards '?' and '*'. Here, '?' represents any *single* character of the text, and '*' represents any substring of the text (including the empty substring). For example, the pattern "A?B*?A" matches the text "ABACBCABBCCACBA" starting in position 3 (in three different ways), and position 7 (in two ways). For this input, your algorithm would need to return '3'.

   UPDATE: You may assume that the pattern you are trying to match containst at most 3 blocks of question marks; the usage of '*' wildcards is stll unrestricted. Here, a block refers to a string of consecutive '?'s in the pattern. For example, AAB??ACA???????BB contains 2 blocks of question marks; A?B?C?A?C contains 4 blocks of question marks.

4. In the two-dimensional pattern-matching problem, you are given an $m \times n$ matrix M and a $p \times q$ pattern P. You wish to find all positions (i, j) in M such that the the submatrix of M between rows i and $i + p - 1$ and between columns j and $j + q - 1$ is identical to P. (That is, the $p \times q$ sub-matrix of M below and to the right of position (i, j) should be identical to P.) Describe and analyze an efficient algorithm to solve this problem.[1]

[1] Note that the normal string-matching problem is the special case of the 2-dimensional problem where $m = p = 1$.