Pattern Matching Algorithms 88-961-01

Moed I March 10, 1999, 12:00

Instructor: *Prof. A. Amir*

Time: Two hours

External materials may be consulted!

Answer all questions.
All questions have equal weight.

1. In most algorithms for two dimensional matching the input is a rectangular pattern. In "life" most patterns are not rectangles.

   (a) Give a method for representing non-rectangular inputs.

   (b) Describe an idea for an algorithm that does exact pattern matching for non-rectangular patterns.

   (c) What is the time complexity of your algorithm?
   (Maximum score will be given for an algorithm whose time complexity is $O(|T| \log^2 |P|)$.)

2. The *inverse pattern matching problem* is defined as follows: Let $T = t_1 \cdots t_n$ be the text and $P = p_1 \cdots p_m$ be the pattern. The *hamming distance* between P and location i in the text is

$$ham(P, T_i) =^{def} \sum_{i=1}^{m} c(p_j, t_{i+j-1})$$

   where

$$c(a, b) = \begin{cases} 0, & \text{if } a = b; \\ 1, & \text{if } a \neq b. \end{cases}$$

   In words, we count the mismatches resulting from matching P with location i in the text.

   The *average hamming distance* between P and T is:

$$ham(P, T) =^{def} \frac{\sum_{i=1}^{n-m+1} ham(P, T_i)}{n - m + 1}.$$

   (a) For input text T and m, $m \leq n$, describe an algorithm that finds a substring P' of T of length m for which $ham(P', T)$ is minimum.

   (b) What is the running time of the algorithm you described in item (a)?

   (c) For input text T and m, $m \leq n$, describe an algorithm that finds a string P (not necessarily a substring of T) of length m, for which $ham(P, T)$ is minimum.

   (d) What is the running time of the algorithm you described in item (c)? (A polynomial time algorithm gives a *bonus* of five points).

3. Consider the *2-d k-mismatches problem* defined below.

   Let Σ be a given finite alphabet.

   INPUT: Two dimensional text $T[n \times n]$ and two dimensional pattern $P[m \times m]$, natural number $k < m^2$.

   OUTPUT: All locations in T where there is an occurrence of P with no more than k mismatches.

   Describe an algorithm for solving the 2-d k-mismatches problem in time $O(kn^2)$.

GOOD LUCK