COMP20007 Design of Algorithms

Week 11 Workshop

Tutorial

1. Counting Sort Use counting sort to sort the following array of characters:

`[a, b, a, a, c, d, a, a, f, c, b]`

How much space is required if the array has n characters and our alphabet has k possible letters?

2. Radix Sort Use radix sort to sort the following strings:

`abc bab cba ccc bbb aac abb bac bcc cab aba`

As a reminder radix sort works on strings of length k by doing k passes of some other (stable) sorting algorithm, each pass sorting by the next most significant element in the string. For example in this case you would first sort by the 3rd character, then the 2nd character and then the 1st character.

3. Stable Counting Sort Which property is required to use counting sort to sort an array of tuples by only the first element, leaving the original order for tuples with the same first element. For example the input may be:

$$(8, \texttt{campbell}),\ (6, \texttt{tal}),\ (3, \texttt{keir}),\ \ldots (6, \texttt{gus}),\ (0, \texttt{nick}),\ (8, \texttt{tom})$$

Discuss how you would ensure that counting sort satisfies this property. Can you achieve this using only arrays? How about using auxiliarry linked data structures?

4. Horspool's Algorithm Use Horspool's algorithm to search for the pattern GORE in the string ALGORITHM.

5. Horspool's Algorithm Continued How many character comparisons will be made by Horspool's algorithm in searching for each of the following patterns it the binary text of one million zeros?

(a) 01001

(b) 00010

(c) 01111

6. Horspool's Worst-Case Time Complextity Using Horspool's method to search in a text of length n for a pattern of length m, what does a worst-case example look like?

7. (Revision) Recurrence Relations Solve the following recurrence relations. Give both a closed form expression in terms of n and a Big-Theta bound.

(a) $T(1) = 1$, $T(n) = T(n/2) + 1$.

(b) $T(0) = 0$, $T(n) = T(n-1) + n/5$.

8. (Optional) Karp-Rabin Hashing In this question we will use Karp-Rabin hashing to solve a string search problem.

For an alphabet with a characters, and some positive number m (we usually select m to be prime, why?) the Karp-Rabin hash function for a string of n characters $S =$"$s_0 s_1 \dots s_{n-1}$" is given by:

$$h(S) = \sum_{i=0}^{n-1} a^{n-1-i} \cdot \text{chr}(s_i) \mod m.$$

Here $\text{chr}(s)$ gives the index of the character s in the alphabet, *i.e.*, $\text{chr}(s) \in \{0, 1, \dots, a-1\}$.

Consider the following example. We have the alphabet { A, B, C, D }, and as such $a = 4$. Let $m = 11$.

We are going to search for the string $P =$"CAB" in the string $T =$"CADACAB".

Since $|P| = 3$ you'll have to compute the hash for each substring of 3 characters in T. To start you off, consider the first subtrings $T[0 \dots 2] =$"CAD",

$$\begin{aligned} h(\text{“CAD”}) &= a^2 \cdot \text{chr(C)} + a \cdot \text{chr(A)} + \text{chr(D)} \mod m \\ &= 4^2 \cdot 2 + 4 \cdot 0 + 3 \mod 11 \\ &= 35 \mod 11 \\ &= 2 \end{aligned}$$

Now we can, in constant time, compute the successive three characters, by using the following formula:

$$h(s_1 s_2 \dots s_k) = a\left(h(s_0 s_2 \dots s_{k-1}) - a^{k-1} \cdot \text{chr}(s_0)\right) + \text{chr}(s_k) \mod m$$

So, we can compute $h($"ADA"$)$ like so:

$$\begin{aligned} h(\text{“ADA”}) &= 4\left(h(\text{“CAD”}) - 4^2 \cdot 2\right) + 0 \mod m \\ &= 4(2 - 32) + 0 \mod 11 \\ &= 4(-30) + 0 \mod 11 \\ &= 4(-30 \mod 11) \mod 11 \\ &= 4(3) \mod 11 \\ &= 12 \mod 11 \\ &= 1 \end{aligned}$$

Complete the string search by computing all of the required hashes (including $h(P)$) and determining whether or not $h(P)$ matches any of the hash values for the substrings of T.

How does this enable us to search for P in T in $O(|T| + |P|)$ time? What might go wrong?