Basic Algorithms, Problem Set 5

Solutions

1. Consider hashing with chaining using as hash function the sum of the numerical values of the letters (A=1,B=2,...,Z=26) mod 7. For example, h(JOE)= 10+15+5mod7 = 2. Starting with an empty table apply the following operations. Show the state of the hash table after each one. (In the case of Search tell what places were examined and in what order.)
   Insert COBB
   Insert RUTH
   Insert ROSE
   Search BUZ
   Insert DOC
   Delete COBB
   `Solution:`Solution:Let T[0 6] be the hash table which is NIL, NIL, NIL, NIL, NIL, NIL, NIL initially. Let num(·) : $\{A, B, , Z\} \rightarrow \{1 \cdots 26\}$ be the specified bijection which maps a letter to its numerical value. We have
   Insert COBB: num(C) + num(O) + num(B) + num(B) mod 7 = (3 + 15 + 2 + 2) mod 7 = 22 mod 7 = 1 T[1] is empty, so COBB is placed in T[1]. T[0 6] = NIL, COBB, NIL, NIL, NIL, NIL, NIL.
   Insert RUTH: num(R) + num(U) + num(T) + num(H) mod 7 = (18 + 21 + 20 + 8) mod 7 = 67 mod 7 = 4 T[4] is empty, so RUTH is placed in T[4]. T[0 6] = NIL, COBB, NIL, NIL, RUTH, NIL, NIL.
   Insert ROSE: num(R) + num(O) + num(S) + num(E) mod 7 = (18 + 15 + 19 + 5) mod 7 = 57 mod 7 = 1 So ROSE is placed as the head of the linked list in T[1]. T = NIL, ROSE→ COBB, NIL, NIL, RUTH, NIL, NIL.
   Search BUZ: num(B) + num(U) + num(Z) mod 7 = (2 + 21 + 26) mod 7 = 49 mod 7 = 0 T[0] is empty, it would not contain BUZ NIL (representing not found) is returned. Hash table T remains unchanged.
   Insert DOC: num(D) + num(O) + num(C) mod 7 = (4 + 15 + 3) mod 7 = 22 mod 7 = 1 So DOC is placed as the head of the linked list in T[1]. T = NIL, DOC→ ROSE→ COBB, NIL, NIL, RUTH, NIL, NIL.
   Delete COBB: As calculated before, the key for COBB is 1. So COBB is fetched in T[1]. After DOC and ROSE are examined, COBB is found and then deleted. T = NIL, DOC→ ROSE, NIL, NIL, RUTH, NIL, NIL.

2. Sunway TaihuLight [1] is tested on Karatsuba's algorithm and it takes 8 minutes to multiply two numbers, each having one billion digits. How long would it take to multiply two numbers, each having eight billion digits. (Assume the addition in Karatsuba's algorithm takes negligible time.)
   `Solution:`Karatsuba makes three calls to multiplication of four billion digit numbers, which in turn make nine calls to multiplication of two billion digit numbers, which in turn make 27 calls to multiplication of one billion digit numbers, so the total time is $27 \cdot 8 = 216$ minutes.

3. **Just For Fun!** Here is a puzzle from *Mathematical Puzzles*, a great new book by Peter Winkler:
   You have 25 horses and can race them in groups of five, but having no stopwatch you can only observe the order of finish. Find the *top three* horses (with their order) with seven races.
   `Solution:`Split the 25 into five groups of five, race each group and then race the winners. Wlog [2] we call label the horses $A1 \cdots 5$;$B1 \cdots 5$, $\cdots$, $E1 \cdots 5$ with the five races going in numerical order (1 the winner) and the sixth race $ABCDE$. So we know A is the overall winner. But now the second and third can only be $A2, A3, B1, B2, C1$. So race them to determine second and third!

4. This problem gives some approaches toward finding the median of n elements.

   (a) Assume a routine `YUE[A]` which takes input $A[1 \cdots n]$ and returns that i so that $A[i]$ is in position precisely $\frac{n}{2} + \sqrt{n}$ when the data is ordered. Assume further [3] that `YUE` takes time $O(n)$. Using `YUE` create an algorithm `MEDIAN[A]` that returns that i such that $A[i]$ is the precise median (suppose n odd) of the data and such that `MEDIAN[A]` takes time $O(n)$. (Or, if you wish, you can make the output the value of the median.) Give the analysis of the time.
   `Solution:`First apply `YUE`, returning i. (Time $O(n)$) Then compare $A[i]$ will all other $A[j]$, putting the $\frac{n}{2} + \sqrt{n}$ (including i itself) values $\leq A[i]$ in an auxilliary array B. Now (as in the previous assignment!) we turn B into a MAXHEAP (Time $O(n)$), EXTRACT-MAX $\sqrt{n}$ times (Time $O(\sqrt{n} \lg n)$ from previous assignment). Now $B[1]$ is the median value of the array A. To get

[1] supercomputer
[2] wlog= without loss of generality
[3] While YUE is fictitious, something similar does exist

the index (there are better ways here!) we run through array A until finding s with $A[s] = B[1]$ (time $O(n)$) and return s. Total time is a bunch of $O(n)$ or smaller terms, so the sum is $O(n)$.

(b) (*) [4] Now suppose instead of `YUE[A]` you had `CARLOS[A]` which takes input $A[1 \cdots n]$ and returns some i so that $A[i]$ is in position between $\frac{n}{2} - \sqrt{n}$ and $\frac{n}{2} + \sqrt{n}$ when the data is ordered and takes time $O(n)$. However, `CARLOS[A]` doesn't tell you precisely where $A[i]$ lies in the ordering. Using `CARLOS` create an algorithm `MEDIAN[A]` that returns that i such that $A[i]$ is the precise median (suppose n odd) of the data and such that `MEDIAN[A]` takes time $O(n)$. (Or, if you wish, you can make the output the value of the median.) Give the analysis of the time.

`Solution:`Apply `CARLOS` returning i. Compare $A[i]$ with all other values, time $O(n)$, thus finding its precise position $\frac{n}{2} + k$ where $|k| \leq \sqrt{n}$. If it is the median we are done. Say it is greater than the median – in position $\frac{n}{2} + k$ with k positive. Create an auxilliary array B of those values $\leq A[i]$. We make B into a MAXHEAP, EXTRACTMAX k times, and then return $B[1]$. If $A[i]$ were smaller than the median, in position $\frac{n}{2} - k$ with k positive, we create an auxilliary array B of those values $\geq A[i]$. We make B into a MINHEAP (it has size $\frac{n}{2} + k$), EXTRACTMIN k times, and then return $B[1]$.

Usually when we hear or read something new, we just compare it to our own ideas. If it is the same, we accept it and say that it is correct. If it is not, we say it is incorrect. In either case, we learn nothing.
Thich Nhat Hanh

[4] (*) represents a more challenging problem but still part of the assignment