University of Waterloo
CS240E, Winter 2023
Assignment 4

Due Date: Wednesday, March 22, 2023 at 5pm

Be sure to read the assignment guidelines (`http://www.student.cs.uwaterloo.ca/~cs240e/w23/guidelines/guidelines.pdf`). Submit your solutions electronically as individual PDF files named a4q1.pdf, a4q2.pdf, ... (one per question).

Question 1 [1+2+2+5=10 marks]

Assume we have a hash function h for some table-size $M \geq 2$, and define a probe sequence as follows:

$$\begin{aligned} h(k,0) &= h(k) \\ h(k,i) &= h(k,i-1) + i \bmod M \quad \text{for } 1 \leq i < M \end{aligned}$$

a) Write the probe sequence for $h(k) = 0$ and $M = 8$ starting from $i = 0$ to $i = M - 1$.

b) Show that this probe sequence is an instance of quadratic probing.

c) Show that if $h(k,i){=}h(k,j)$ for some $0 \leq i < j < M$, then $(j{-}i)(j{+}i{+}1){=}0 \bmod 2M$.

d) Assume that M is a power of 2, say $M = 2^m$ for some integer m. Prove that all entries in the probe sequence are different, therefore the probe sequence will hit an empty slot.

Question 2 [2+4+5 = 11 marks]

We have seen one method of obtaining a universal family of hash-functions in class. This assignment discusses another one. Let us assume that all keys come from some universe $\{0, \ldots, U-1\}$, where $U = 2^u$. Therefore any key k can be viewed as bitstring x_k of length u by taking its base-2 representation.

Let us assume further that the hash-table-size M is $M = 2^m$ for some integer m, with $m < u$. To choose a hash-function, we now randomly choose each entry in a $m \times u$-matrix H to be 0 or 1 (equally likely). Then compute $h_k = (Hx_k)\%2$, where x_k is now viewed as a vector and '%2' is applied to each entry. The output is a m-dimensional vector with entries in $\{0,1\}$; interpreting it as a length-m bitstring gives a number $\{0, ..., M-1\}$ that we use as hash-value $h(k)$. For example, if $k = 18$, $u = 5$, $m = 3$ and H is as shown below, then

$h(k) = 1$ since

$$\underbrace{\begin{pmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}}_{H} \underbrace{\begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}}_{\text{18 as length-5 bitstring}} \%2 = \underbrace{\begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix}}_{Hx_k} \%2 = \underbrace{\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}}_{\text{1 as length-3 bitstring}}$$

a) Let H be the above matrix, $u = 5$ and $m = 3$. Consider the keys 9 and 13. What are their hash-values (as numbers in $\{0, \ldots, M-1\}$? Show your work.

b) Consider again $u = 5, m = 3$ and keys $k = 9$ and $k' = 13$. Consider the same matrix H, except that the bits in the third column are randomly chosen. What is the probability that $h(k) = h(k')$? Justify your answer.

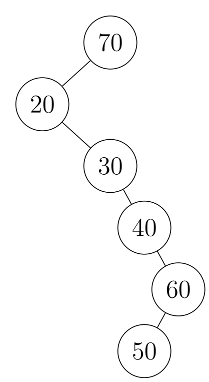
c) Assume now that all of H is chosen randomly and independently. Show that (for any u, m) this gives a universal hash function family, or in other words, $P(h(k) = h(k')) \leq \frac{1}{M}$ for any two keys $k \neq k'$.

d) [Possibly graded, 2 marks] This method for obtaining universal hash-functions is much less popular than using the Carter-Wegman functions. Why do you think that that might be the case? (Expected length of answer is 1-3 sentences.)

Question 3 [1+2+9+5+4=21 marks]

This assignment asks you to compare the performance of the MTF-heuristic for binary search trees with splay trees.

a) Consider the binary search tree shown on the right.



  i) What is its potential function value when viewed as a splay tree? (State it with two fractional digits.)

  ii) Show the binary search tree that results if you perform *splayTree::search*(50).

  For both part-questions, it suffices to state the correct final answer but we recommend showing some intermediate steps so we can give part-marks in case of errors.

b) Let T be a binary search tree with n nodes and height $h = n - 1$, i.e., T is a path from the root to a unique leaf x. Show that if we perform *splayTree::search*(k) for the key k at x, then the resulting tree T' has height at most $h/2 + c$ for some constant c. Make c as small as possible.

Hint: Show a bound on the height of the subtree rooted at x after you have done i operations.

c) Create an example of a binary search tree T with n nodes and a sequence of $\Theta(n)$ operations *BST-MTF::search* for keys in T such that the total number of rotations is in $\Theta(n^2)$.

d) Prof. I.N.Correct claims that for any n they have an example of a binary search tree T with n nodes and a sequence of n operations *SplayTree::search* for keys in T such that the total number of rotations is in $\Theta(n^2)$. In particular the actual run-time for these n operations is in $\Omega(n^2)$.

Prove that this is impossible.

Question 4 [3 marks]

Recall *interpolation-search* (Algorithm 6.3 from the course notes) and consider its performance for the sorted array $A[0..n-1]$ where $A[i] = ai + b$ for $0 \leq i \leq n-1$ (for some constants $a > 0$ and b that are arbitrary real numbers). Show that then a search for a key k always takes $O(1)$ time, regardless of whether key k is in A or not.

Question 5 [8 marks]

This question concerns sorting a set of infinite-precision numbers $x_0, \ldots, x_{n-1}$. Specifically, each x_i is in $[0, 1)$ and written in base-2. It is given to you implicitly, via an accessor-function *get-decimal-place*(i, d), which returns the bit in the dth decimal place of x_i. For example, if $x_i = 0.001001...$ then *get-decimal-place*$(i, 3) = 1$ and *get-decimal-place*$(i, 4) = 0$. Function *get-decimal-place* takes $\Theta(1)$ time.

Describe an algorithm to sort these (implicitly given) numbers $x_0, \ldots, x_{n-1}$ in $O(n \log n)$ expected time, assuming the numbers $x_0, \ldots, x_{n-1}$ have been randomly and uniformly chosen from the interval $[0, 1)$. You may also assume that all numbers are distinct. Note that comparing x_i and x_j is *not* a constant-time operation! Your output should be the sorting-permutation π (i.e., $x_{\pi(0)} < x_{\pi(1)} < \cdots < x_{\pi(n-1)}$).

A high-level description is enough, no need for pseudo-code, and the correctness can be extremely short. (But do argue the run-time carefully.)

Question 6 [moved to A5]

This question is moved to Assignment 5. It should not be submitted to A4 MarkUs.