University of Waterloo
CS240 - Spring 2026
Assignment 4

Due Date: Tuesday July 14, 5pm

Please read the following link for guidelines on submission:

`https://student.cs.uwaterloo.ca/~cs240/assignments.phtml#guidelines`

Late Policy: Assignments are due at 5:00pm, with a grace period until 7:59pm.

Question 1 Analysis of search in skip lists [2+1+5+3 marks]

We work with a skip list with n keys, and we consider the search for a key k. This key, and the keys in the skip list, are fixed throughout. All the expected values are computed with respect to the coin tosses we do to build the skip list. For $i \geq 0$, we call N_i the number of nodes we visit at level i when searching for k (this is a random variable, it depends on the skip list).

(a) Using the expected value of the number of nodes at level i seen in class, give a simple upper bound on $E[N_i]$ that depends on i.

(b) Justify the following observation: if we follow a search path backward, then whenever we can go up, we must go up.

(c) For a level $i \geq 0$, let $1 \leq \ell_i \leq r_i \leq n$ be the indices of the towers that we visit at level i during the search. Prove that, for a given value of r_i, we have

- $P(\ell_i = r_i) = 1/2$
- $P(\ell_i = r_i - 1) = 1/4$
- $\ldots$
- $P(\ell_i = 2) = 1/2^{r_i - 1}$
- $P(\ell_i = 1) = 1/2^{r_i - 1}$

Deduce that, still for a given r_i, $E[r_i - \ell_i + 1] \leq 2$, and from that that $E[N_i] \leq 2$.

(d) Use the two bounds on $E[N_i]$ to prove that the total number of nodes we visit during a search is, in expectation, $O(\log n)$. You can assume that n is a power of 2.

Results you can use

- $\sum_{j=0}^{N-1} j/2^j = 2 - (N+1)/2^{N-1}$
- law of total expectation: if X, Y are two random variables, and for any k we have $E[X - Y | X = k] \leq c$, then $E[X - Y] \leq c$.

Question 2 Analysis of self-organizing search [4+2+1+4+3 marks]

In this problem, we analyse the move-to-front strategy for linear search. Suppose we have a linked list with keys $x_1, \ldots, x_n$, and that the probability of accessing x_i is p_i; you can assume that $p_i > 0$ for all i.

We consider the move-to-front heuristic. As input, we suppose we are given a linked list with keys $x_1, \ldots, x_n$, but we do not know in which order they are.

(a) We do not know the order at the beginning, so let us call $X_{i,j}^{(0)}$ the probability that initially x_i is before x_j. Show that after one query, the probability that x_i is before x_j is

$$X_{i,j}^{(1)} = p_i + (1 - p_i - p_j)X_{i,j}^{(0)}$$

and the probability that x_j is before x_i is $1 - X_{i,j}^{(1)}$.

(b) Show by induction that after N queries, these probabilities are

$$X_{i,j}^{(N)} = p_i(1 + (1 - p_i - p_j) + (1 - p_i - p_j)^2 + \cdots + (1 - p_i - p_j)^{N-1}) + (1 - p_i - p_j)^N X_{i,j}^{(0)}$$

and $1 - X_{i,j}^{(N)}$.

(c) Show that the limit probabilities, for $N \to \infty$, are

$$\frac{p_i}{p_i + p_j} \quad \text{and} \quad \frac{p_j}{p_i + p_j}.$$

You can use the fact that for $0 \leq r < 1$, $1 + r + r^2 + r^3 + \cdots = \frac{1}{1-r}$. In the rest of the problem, we will assume that N is very large, and work with the limit probabilities.

(d) Show that for $N \to \infty$ the expected position of x_i in the list is

$$1 + \sum_{j \neq i} \frac{p_j}{p_i + p_j}.$$

Hint: rewrite the number of x_j's before x_i as a sum involving indicator variables. Note: we index positions starting from 1, just as we did in the move-to-front lecture.

(e) What is the expected number C_{MTF} of links visited in a search using this heuristic? (you do not have to simplify the expression). Compute this value for the frequencies of the example seen in class:

key	A	B	C	D	E
access-probability	$\frac{2}{26}$	$\frac{8}{26}$	$\frac{1}{26}$	$\frac{10}{26}$	$\frac{5}{26}$

Question 3 Interpolation search [4+4 marks]

In this problem, to simplify your calculations, assume that the index m we use in interpolation search (step 5) is

$$m = \ell + \left\lfloor \frac{k - A[\ell]}{A[r] - A[\ell]}(r - \ell) \right\rfloor .$$

The formula used in class is better suited for the average case analysis, the one above works better for this problem.

(a) Suppose that we use interpolation search in an array with entries $A[i] = \alpha i$ for $i = 0, \ldots, n-1$, α positive constant. What is the worst-case runtime for search? Give (and justify) a Θ bound.

(b) Suppose that we use interpolation search in an array with entries $A[i] = \sqrt{i}$, for $i = 0, \ldots, n-1$, and that we search for $k = 1$. What is the runtime? Give (and justify) a big-O bound. You can (but don't have to)

- use any result seen in class without reproving them,
- assume that this runtime is an increasing function of n,
- use a sloppy recurrence,
- solve this recurrence only for some special values of n.

The following exercises are provided for additional practice and are not part of the graded portion of the assignment. While these questions will not be evaluated, we strongly encourage you, and expect you, to work through them to deepen your understanding of the material. You are welcome to upload your solutions to Crowdmark for your own record-keeping and organization; however, no grades or feedback will be provided for these submissions. You may also ask questions about these exercises during office hours or on Piazza.

Question 4 Skip lists

Suppose you have a skip list with only three levels. The lower one has $n+2$ entries $-\infty < a_0 < \cdots < a_{n-1} < +\infty$. The middle one has $k+2$ entries, where k is an integer that divides n (so that $n = km$ for some integer m); we assume that (with the exceptions of $\pm\infty$), these entries are evenly spread out, so they correspond to $-\infty, a_0, a_m, a_{2m}, \ldots, a_{(k-1)m}, +\infty$. The top level holds $-\infty, +\infty$.

(a) What is the worst time for a query? Give a $\Theta(\ )$ expression involving k and n, and justify your answer.

(b) Given n, how to choose k so as to minimize the $\Theta(\ )$ bound you just obtained, and what does the worst case become in that case? Give a $\Theta(\ )$ expression in terms of n, and justify your answer. You can assume n is a square.

Question 5 Tries

(a) For $n \geq 1$, find the height of the trie that stores the binary representation of all integers in $\{0, \ldots, 2^n - 1\}$ (without unnecessary leading zeros), that is, $0\$, 1\$, 10\$, \ldots$.

(b) What is the height of the compressed trie storing these words?

Question 6 Hashing

(a) Consider a hash table dictionary with universe $U = \{0, 1, 2, \ldots, 24\}$ and size $M = 5$. If items with keys $k = 21, 3, 16, 1$ are inserted in that order, draw the resulting hash table if we resolve collisions using:

- Linear probing with $h(k) = (k + 1) \mod 5$
- Cuckoo hashing with $h_1(k) = k \mod 5$ and $h_2(k) = \lfloor k/5 \rfloor$ (use two arrays of size 5)

(b) Let S be a set of n keys mapped to a hash table also of size n using chaining. We make the uniform hashing assumption, and we call c_n the expected number of empty slots. Prove that $c_n = n/e + o(n)$, where e is the basis of the natural logarithm.

You can use the fact that $\lim_{n\to\infty}(1 - 1/n)^n = 1/e$. Hint: use indicator variables $I_0, \ldots, I_{n-1}$, that take values 0 or 1, depending on whether the corresponding entry in the table is empty or not; then, find the expected value of each I_i.