XJTLU | Computing

CPT108: Data Structures and Algorithms

Semester 2, 2023-24

Tutorial 3

Hashtables

Problem 1. Insert the keys `E A S Y Q U T I O N` in that order into an initially empty hashtable of $M = 5$ and hash function $h(k) = (11k + 3)\%5$ using separate chaining. (Assume `A` $= 1$, `B` $= 2$, `C` $= 3, \ldots$)

Problem 2. Conisder inserting the keys $10, 22, 31, 4, 15, 28, 17, 88, 59$ into a hashtable of length $m = 11$ using open addressing. Illustrate the results of these keys using (i) linear probing with $h(k, i) = (k + i) \mod m$, and (ii) double hashing with $h_1(k) = k$ and $h_2(k) = 1 + (k \mod (m - 1))$. [Cormen 11.4-1]

Problem 3. Which of the following hash functions will distribute keys most uniformly over 10 buckets number 0 to 9 for k ranging from 0 to 2020?

(*Hint:* You may want to create a spreadsheet, calculate and see how the distribution of the computed hash values look like)

A. $h(k) = (12 \times k) \mod 10$

B. $h(k) = (11 \times k^2) \mod 10$

C. $h(k) = k^3 \mod 10$

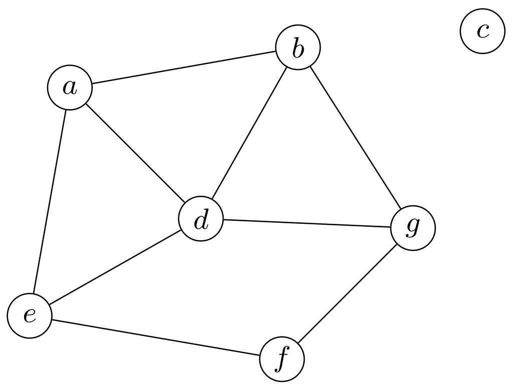
D. $h(k) = k^2 \mod 10$

Briefly explain your answers.

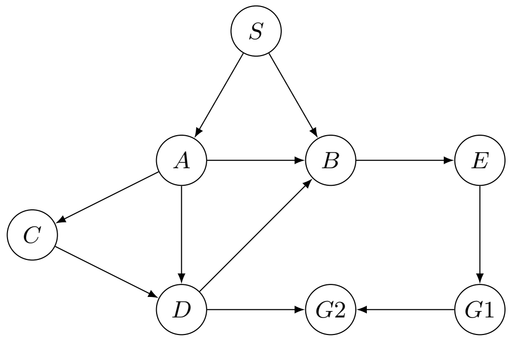
Graphs

Problem 4. What is the maximum number of edges in a graph with V vertices? What is the minimum number of edges in a graph with V vertices, none of which are isolated? [Sedgewick 4.1.1]

Problem 5. Draw the adjacency matrix and adjacency list of the graph below.



Problem 6. Suppose you have the following *directed* graph.



Start from node S, write down the path generated if breadth first search (BFS) is used. Break all ties by picking the nodes in alphabetic order.

Problem 7. Suppose you use a stack instead of a queue when running breadth first search (BFS). Does it still compute shortest paths?

References

Cormen, Thomas H. et al. (2022). *Introduction to Algorithms*. 4th. MIT Press. ISBN: 9780262046305. URL: `https://mitpress.mit.edu/9780262046305`.

Sedgewick, Robert and Kevin Wayne (2011). *Algorithms*. 4th. Addison Wesley.

Solutions

1.

key		$h(k) = 11k + 3$	$h(k)\%5$
E	5	58	3
A	1	14	4
S	19	212	2
Y	25	278	3
Q	17	190	0
U	21	234	4
T	20	223	3
I	9	102	2
O	15	168	3
N	14	157	2

$$\begin{tikzcd}
\boxed{0} \arrow[r] & \boxed{Q} & & & \\
\boxed{1} & & & & \\
\boxed{2} \arrow[r] & \boxed{S} \arrow[r] & \boxed{I} \arrow[r] & \boxed{N} & \\
\boxed{3} \arrow[r] & \boxed{E} \arrow[r] & \boxed{Y} \arrow[r] & \boxed{T} \arrow[r] & \boxed{O} \\
\boxed{4} \arrow[r] & \boxed{A} \arrow[r] & \boxed{U} & &
\end{tikzcd}$$

2. **Linear probing**

key	$h(k) = k$	$h(k)\,\%\,11$
10	10	10
22	22	0
31	31	9
4	4	4
15	15	4
28	28	6
17	17	6
88	88	0
59	59	4

T	Value
0	② 22
1	⑧ 88
2	
3	
4	④ 4
5	⑤ 15
6	⑥ 28
7	⑦ 17
8	⑨ 59
9	③ 31
10	① 10

Double hashing

key	$h_1(k) = k$	$h_2(k) = 1 + (k \mod (m-1))$	$h(k,i) = h_1(k) + i \times h_2(k)$					$h(k,i)\%11$					key
			0	1	2	3	4	0	1	2	3	4	
10	10	1	10	11	12	13	14	10	0	1	2	3	10
22	22	3	22	25	28	31	34	0	3	6	9	1	22
31	31	2	31	33	35	37	39	9	0	2	4	6	31
4	4	5	4	9	14	19	24	4	9	3	8	2	4
15	15	6	15	21	27	33	39	4	10	5	0	6	15
28	28	9	28	37	46	55	64	6	4	2	0	9	28
17	17	8	17	25	33	41	49	6	3	0	8	5	17
88	88	9	88	97	106	115	124	0	9	7	5	3	88
59	59	10	59	69	79	89	99	4	3	2	1	0	59

T	Value
0	② 22
1	
2	⑨ 59
3	⑦ 17
4	④ 4
5	⑤ 15
6	⑥ 28
7	⑧ 88
8	
9	③ 31
10	① 10

3. Since modulo 10 is used, the last digit matters.

   The table on the below shows all numbers from 0 to 9, and their squared and cubic values before and after modulo operation.

Number, k	$12 \times k$	Last digit in $12 \times k$	k^2	Last digit in k^2	$11 \times k^2$	Last digit in $11 \times k^2$	k^3	Last digit in k^3
0	0	0	0	0	0	0	0	0
1	12	2	1	1	11	1	1	1
2	24	4	4	4	44	4	8	8
3	36	6	9	9	99	9	27	7
4	48	8	16	6	176	6	64	4
5	60	0	25	5	275	5	125	5
6	72	2	36	6	396	6	216	6
7	84	4	49	9	539	9	343	3
8	96	6	64	4	704	4	512	2
9	108	8	81	1	891	1	729	9

   As can be seen, all numbers from 0 to 2020 are evenly distributed in the 10 buckets in k^3, but not in the case of k^2 where $2, 3, 7$ and 8 would be empty.

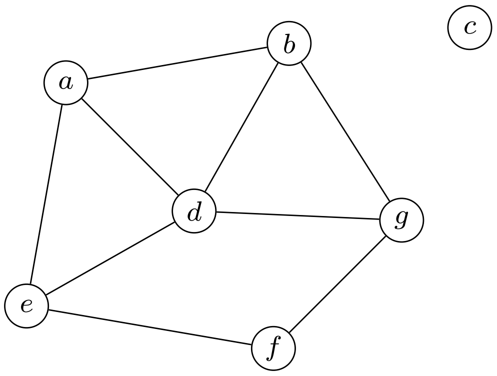
   And similar results can be observed from the other two cases.

   Hence, only option (C) $h(k) = k^3 \mod 10$ cover all the digits from 0 to 9 and is the best answer for this scenario.

4. (a) Maximum number of edges in a fully connected graph $= (n-1) + (n-2) + \cdots + 1$

$$= \frac{(n)(n-1)}{2}$$

   (b) Minimum number of edges in a connected graph $= n - 1$ (same as a linked list)

5. 

	a	b	c	d	e	f	g
a	0	1	0	1	1	0	0
b	1	0	0	1	0	0	1
c	0	0	0	0	0	0	0
d	1	1	0	0	1	0	1
e	1	0	0	1	0	1	0
f	0	0	0	0	1	0	1
g	0	1	0	1	0	1	0

Adjacency matrix

$a \longrightarrow b \quad d \quad e$

$b \longrightarrow a \quad d \quad g$

$c \longrightarrow \emptyset$

$d \longrightarrow a \quad b \quad e \quad g$

$e \longrightarrow a \quad d \quad f$

$f \longrightarrow e \quad g$

Adjacency list

6. $S, A, B, C, D, E, G2, G1$

7. If a stack is used when running a BFS, we will have DFS, which will pick one path, and go in it as deep as it can, then pick another path and so on so forth, which is not optimal for finding the shortest path. This is because a node A could either be reach in 5 steps if you go from one path, or in 10 steps if you go from another path. The DFS might pick the long path and then keep going into it until it find s the node and tells you that this is the path to the node, even though there is a much shorter path that the DFS did not explore yet. We have to continue the set (of results) until the shortest path has been found, but this is too much time wasted.

   It should be noted that we can simulate the function of a queue using *two* stacks (which has been discussed in the lecture). In this situation, the same result of BFS can be computed, but a significant amount of execution time will be used to manipulate the data between the queues.