CSC 3110
Homework 5
Gq5426
May Wandyez

# CSC 3110

## Algorithm Design and Analysis

(30 points)

Due: 04/01/2024 11:59 PM

**Note: Submit answers in PDF document format. Please read the submission format for appropriate file naming conventions.**

1) Exercises 7.1:
   a. Problem 4 (2 points)

**4.** Is the distribution-counting algorithm stable?

Yes, distribution counting algorithm is stable. It is stable because A stable algorithm preserves key order even when the keys are equal. The distribution counting algorithm moves from right to left, and puts equal elements into a sorted array that is also built right to left – meaning that elements remain in the same

distinct integers from 1 to $n$.

**6.** The ***ancestry problem*** asks to determine whether a vertex $u$ is an ancestor of vertex $v$ in a given binary (or, more generally, rooted ordered) tree of $n$ vertices. Design a $O(n)$ input-enhancement algorithm that provides sufficient information to solve this problem for any pair of the tree's vertices in constant time.

b.
Preprocessing the information allows the graph to run faster. The first thing to do is calculate the lowest common ancestor of two vertices using the parent information. The lowest common ancestor is the first ancestor of u that is also an ancestor of v. This involves finding all ancestors of v, then starting with u and going through its ancestors and finding the first shared one. If u is the LCA of v that means u is the ancestor of v. This solution is linear in preprocessing data because each node only needs to be considered once, as LCA can be done concurrently for multiple nodes since they are all part of the same tree. Once the LCA information is processed for each node, it can then be run in constant time to check if the LCA is shared or not.

2) Exercises 7.2:

CSC 3110
Homework 5
Gq5426
May Wandyez

a. Problem 2 (part a & b) (2 points)

BESS_KNEW_ABOUT_BAOBABS

**2.** Consider the problem of searching for genes in DNA sequences using Horspool's algorithm. A DNA sequence is represented by a text on the alphabet {A, C, G, T}, and the gene or gene segment is the pattern.

**a.** Construct the shift table for the following gene segment of your chromosome 10:

TCCTATTCTT

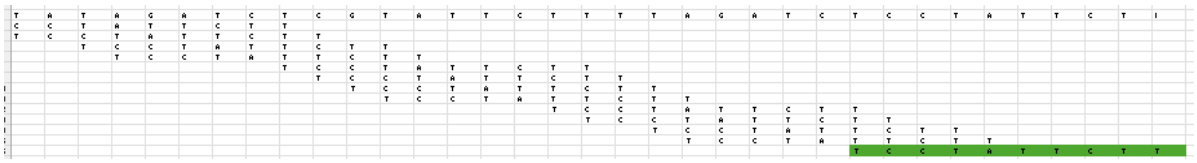
**b.** Apply Horspool's algorithm to locate the above pattern in the following DNA sequence:

TTATAGATCTCGTATTCTTTTATAGATCTCCTATTCTT

a.)

| A | C | G | T |
|---|---|---|---|
| 5 | 2 | 10 | 1 |

b.



Wow look at that excel spreadsheet screenshot, zoom in really far, it was captured on a 4K screen.

c. Problem 8 (part a & b) (2 points)

**8. a.** Would the Boyer-Moore algorithm work correctly with just the bad-symbol table to guide pattern shifts?

**b.** Would the Boyer-Moore algorithm work correctly with just the good-suffix table to guide pattern shifts?

A.) Yes, Boyer-Moore with just the bad-symbol table will work because it's just Horspool's algorithm.
B.) No, Boyer Moore needs the bad-symbol table to work, otherwise the algorithm cannot identify characters that do not match.

3) Exercises 7.3:

CSC 3110
Homework 5
Gq5426
May Wandyez

a. Problem 1 (parts a – c) (2 points)

**1.** For the input 30, 20, 56, 75, 31, 19 and hash function $h(K) = K \bmod 11$
- **a.** construct the open hash table.
- **b.** find the largest number of key comparisons in a successful search in this table.
- **c.** find the average number of key comparisons in a successful search in this table.

a.)Step 1: Apply the hash function h(K) = k mod 11

| 30 | 20 | 56 | 75 | 31 | 19 |
|---|---|---|---|---|---|
| 8 | 9 | 1 | 9 | 9 | 8 |

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| | 56 | | | | | | | 30 | 20 |
| | | | | | | | | 19 | 75 |
| | | | | | | | | | 31 |

Look at that clustering. Much wow.

b.)Because there is a linked list of 3, the largest search will require traversing the entire linked list, so the largest number of key comparison for search is 3.
c.)Ugg averages. 1/6 + 2/6 + 2/6 + 1/6 + 2/6 + 3/6 = 1.6666666666667

b. Problem 5 (2 points)

**5.** *Birthday paradox* The birthday paradox asks how many people should be in a room so that the chances are better than even that two of them will have the same birthday (month and day). Find the quite unexpected answer to this problem. What implication for hashing does this result have?

This is known sometimes as the "Birthday Attack" problem. The answer is 22.5. The implication for hashing is that even when the size of a hash is seemingly massive, there still tends to be clustering from collisions even in small data sizes. Collisions are unavoidable – you just have to live with them and minimize them.

4) Exercises 7.4:

a. Problem 3 (2 points)

**3.** Find the minimum order of the B-tree that guarantees that the number of disk accesses in searching in a file of 100 million records does not exceed 3. Assume that the root's page is stored in main memory.

We need a B tree height that is 3 to reduce the number of searches to 3. Which means we need the m value for that B tree with height 3 and nodes =

100,000,000. B tree height is calculating using the following equation
log(m/2)((n+1)/4) = log(m/2)(( 100,000,001/4)) = log(m/2)( 25000000.25) = 3

(m/2)^3 = 25000000.25
m/2 = 292.41
m = 584.82 rounded up to 585 because we can't have a fraction of a child.

b. Problem 4 (2 points)

**4.** Draw the B-tree obtained after inserting 30 and then 31 in the B-tree in Figure 7.8. Assume that a leaf cannot contain more than three items.

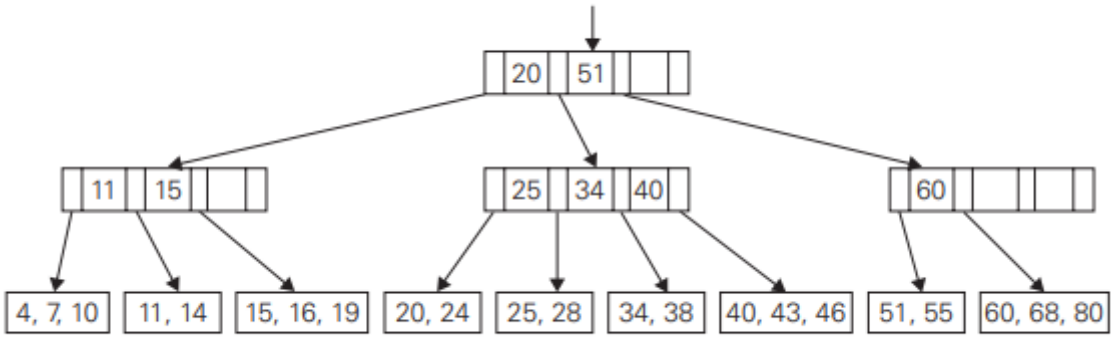
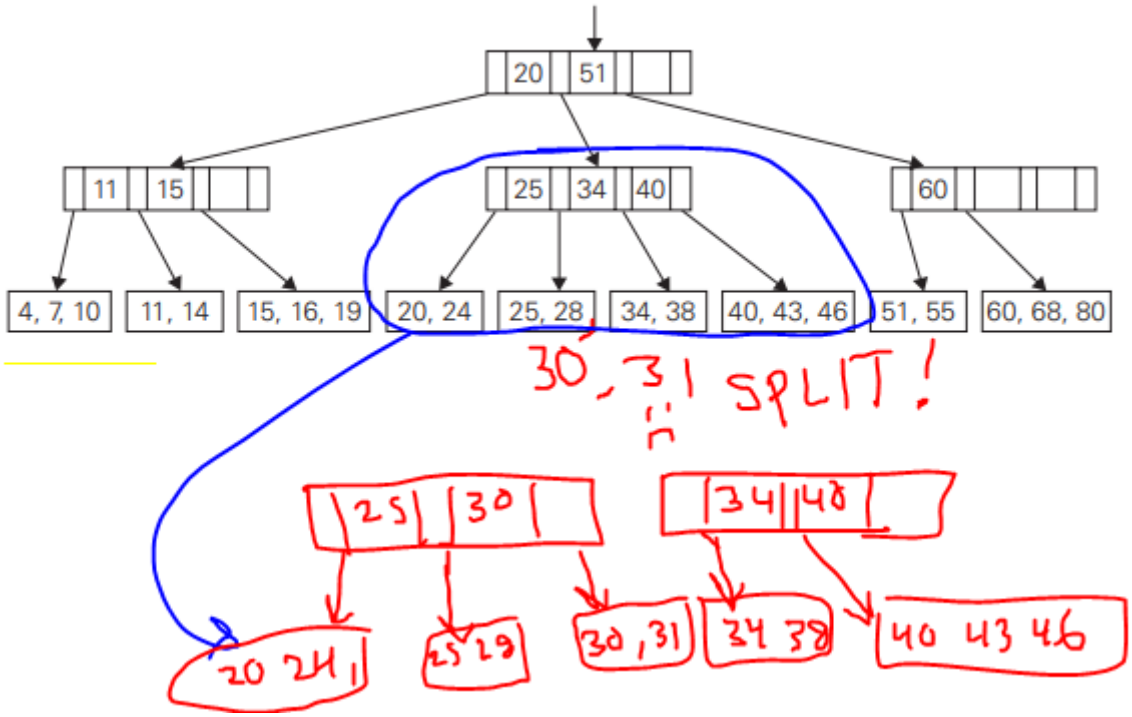



**FIGURE 7.8** Example of a B-tree of order 4.




5) Exercises 8.1

a. Problem 1 (1 point)

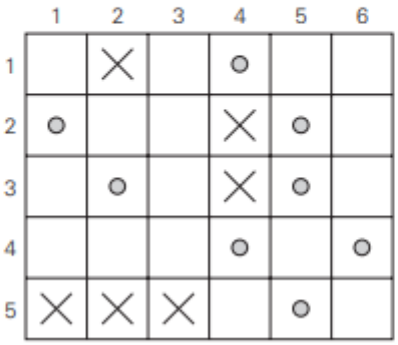
**1.** What does dynamic programming have in common with divide-and-conquer? What is a principal difference between them?

Dynamic programming and divide and conquer both involve breaking down

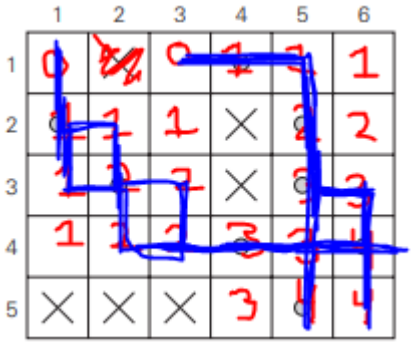
the larger problem into smaller subproblems. Dynamic programming however is built around the subproblems overlapping and reusing their results (often through use of a recursive function), divide and conquer involves there being no overlap between subproblems at all.

b. Problem 5 (2 points)

**5.** How would you modify the dynamic programming algorithm for the coin-collecting problem if some cells on the board are inaccessible for the robot? Apply your algorithm to the board below, where the inaccessible cells are shown by X's. How many optimal paths are there for this board?




If the cell is admissible, the function does not consider it. It is as simple as that.


There are at least 14 optimal paths.

6) Exercises 8.2

a. Problem 4 (part a & b) (2 points)

**4. a.** True or false: A sequence of values in a row of the dynamic programming table for the knapsack problem is always nondecreasing?

**b.** True or false: A sequence of values in a column of the dynamic programming table for the knapsack problem is always nondecreasing?

a.) True. The way that the dynamic programming solution is built for the knapsack problem means that it is based on the maximum of the values to the cells to the left and above it, as a result the values always increase.

b.)Also True, because the value of the knapsack problem is based on adjacent cells, this means that the knapsack is always increasing in value when descending as well.

CSC 3110
Homework 5
Gq5426
May Wandyez

b. Problem 5 (2 points)

**5.** Design a dynamic programming algorithm for the version of the knapsack problem in which there are unlimited quantities of copies for each of the $n$ item kinds given. Indicate the time efficiency of the algorithm.

a.) Divide each of the items value by their weight
b.) Pick the item that has the maximum value per weight
c.) Put as many of that item inside of the knapsack, when no more can be placed, try whichever has the next highest value per weight, see if it fits, repeat until all items are considered

This algorithm will usually generate the optimum value to place in the knapsack, and more importantly it does so incredibly quickly, at a time efficiency of O(i), this linear efficiency is worth the lack of optimality, as in most situations uniformity of good loaded into the knapsack outweighs the benefit of potential additional value.

7) Exercises 8.3

a. Problem 3 (2 points)

**3.** Write pseudocode for a linear-time algorithm that generates the optimal binary search tree from the root table.

Optimal binary search tree is kinda vague, optimal in terms of search? Here is a formula that will build

The optimal binary search tree is built by placing the highest frequency searches at the highest points of the tree, this guarantees that tree is searched through as quickly as possible.

b. Problem 9 (1 point)

**9.** Generalize the optimal binary search algorithm by taking into account unsuccessful searches.

8) Exercises 8.4

a. Problem 3 (2 points)

**3.** Explain how to implement Warshall's algorithm without using extra memory for storing elements of the algorithm's intermediate matrices.

There is a way to update Warshall's algorithm by doing the calculations in place instead of writing temporary matrices. This change involves altering the lowest subloop of Warshall's algorithm to consider if the path through the highest iterative value loop is shorter than the existing point in the graph. If it is, then it is replaced. Because the highest loop passes through every vertex,

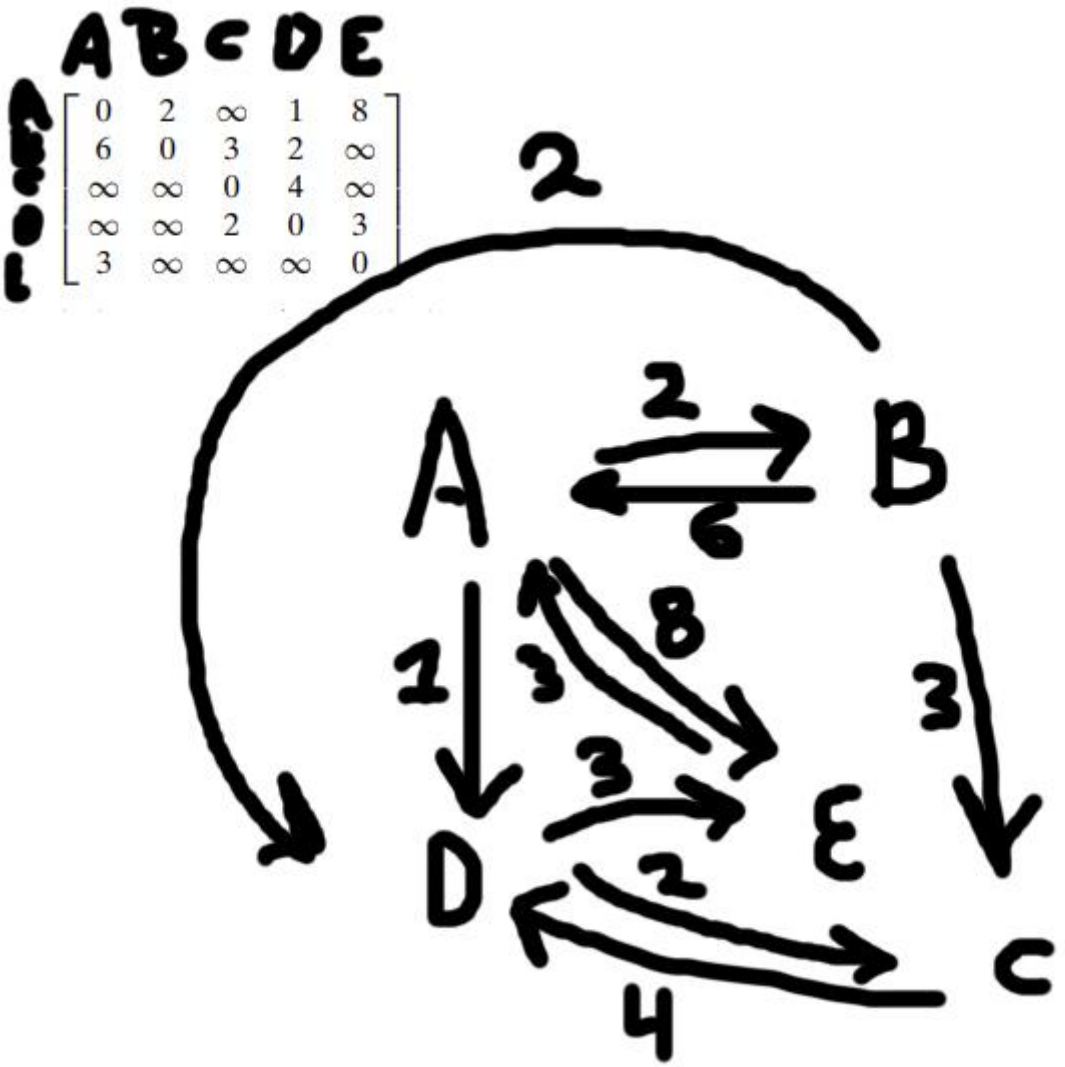
this means that every additional path is considered, and only the shortest path is kept.

b. Problem 7 (2 points)

**7.** Solve the all-pairs shortest-path problem for the digraph with the following weight matrix:

$$\begin{bmatrix} 0 & 2 & \infty & 1 & 8 \\ 6 & 0 & 3 & 2 & \infty \\ \infty & \infty & 0 & 4 & \infty \\ \infty & \infty & 2 & 0 & 3 \\ 3 & \infty & \infty & \infty & 0 \end{bmatrix}$$

Why use a fancy algorithm when you can just draw it by hand and look really hard? Is this fancy and scalable? NO IT IS NOT. Is it the faster than writing down matrix after matrix? YES IT IS. Considering I am more concerned about time efficiency than scalability, this is the best solution.




| | | | | |
|---|---|---|---|---|
| 0 | 2 | 3 | 1 | 4 |
| 6 | 0 | 3 | 2 | 5 |

CSC 3110
Homework 5
Gq5426
May Wandyez

| | | | | |
|---|---|---|---|---|
| 10 | 12 | 0 | 4 | 7 |
| 6 | 8 | 2 | 0 | 3 |
| 3 | 5 | 6 | 4 | 0 |