Ph.D. Comprehensive Examination

Computer Science Department
University of Miami

August 14, 2020

Student Name: Daniel Verdear Student Number: C11489288

Problem number	Points (10 max)
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
Total:	

1. **I.A Data organization; III.A Algorithms and complexity**

   Each item x in a set S has a unique key $key[x]$. We need to implement the following operations.

   (a) Search (S, key)
   (b) Insert (S, x)
   (c) Successor (S, x)
   (d) Predecessor (S, x)

   Give the 4 running times as an O() for the following implementations:

   (a) Ordered (sorted) array,
   (b) Ordered doubly linked list,
   (c) Min-Heap, and
   (d) Hash table

Ordered array

(a) Search (S, key) will run in $O(\log n)$ time because the function can implement a binary search

(b) Insert (S, x) will run in $O(n)$ time because the Function will have to shift all elements greater than x down the array

(c and d) Predecessor and Successor will run in $O(1)$ time IF the key/array index is known. Otherwise, it must be found through by calling search which will take $O(\log n + 1) = O(\log n)$ time.

DLL

(a) Search will run in $O(n)$ time because the list must be traversed in order to carry out a search.

(b) Insert will run in $O(n)$ time because the location of x must first be found by calling Search, then the pointers are changed to reference x. $O(n+1) = O(n)$ time

(c and d) Predecessor and Successor will run in $O(1)$ time because x contains a reference pointer to its predecessor and successor. Accessing these pointers directly will yield the result.

Min-Heap

(a) Search will run in $O(n)$ time because unlike a binary search tree, a min-heap has no guarantee that left_child < right_child. Therefore, it is likely that search will have to traverse the full heap.

(b) Insert will run in $O(\log n)$ time because a min-heap places x at the bottom of the heap and swaps it with its parent if $x <$ parent. Since the heap will have depth of $\log n$ on average, $\log n$ swaps will be performed.

(c and d) Predecessor and Successor will run in $O(n)$ time because there are no guarantees about the local structure of a min-heap. A parent will be less than a child, but there is no guarantee that the parent is the direct predecessor. Therefore, Predecessor and Successor will resolve to traversal.

Hash Table

(a) Search will run in $O(1)$ time because the function will simply calculate the hash value of x. That key will contain x or it will not.

(b) Similarly, Insert will run in $O(1)$ time because the function will calculate the hash value of x and place x in the corresponding table entry.

(c and d) Predecessor and Successor will run in $O(n)$ time because the hash function should behave randomly. The information from x's hash value cannot provide information on nearby values' hash value. Therefore, Predecessor and Successor will resolve to traversal.

2. **I.B Program control and structure; I.C Programming language and notations**

Suppose that procedure *swap* is declared as follows:

```
procedure swap( x, y: integer);
    procedure f(): integer;
        var z: integer;
        begin // f
            z = x; x = y; return z;
        end // f
    begin // swap
        y = f();
    end // swap
```

Describe the effect of the procedure call *swap*(i, A[i]) under each of the following parameter passing methods:

(a) Call-by-value → copy value into local parameter

(b) Call-by-reference → pass the access path to parameter

(c) Call-by-value-result → pass the access path to parameter AND returned-to variable

(a) i and A[i] (values) are copied into x and y.

f() is called and swaps y into x, then returns x into y.

x is copied into the i variable and y is copied into the A[i] variable, swapping them.

(b) x is given a pointer to i; y is given a pointer to A[i].

f() is called; x is assigned the pointer to A[i] and y is returned a pointer to i.

Swap returns and i is given a pointer to A[i]; A[i] is given a pointer to i.

(c) x is given a pointer to i; y is given a pointer to A[i]

f() is called; x is assigned the pointer to y and y is returned a pointer to z. (which contains the pointer to i)

Swap returns and i is given a pointer to x and A[i] is given a pointer to y.

3. **I.D Software engineering**

From the software engineering point of view, any software development process can be divided into several sub-disciplines:

(a) Requirement Analysis
(b) Functional Specification
(c) Architectural Design
(d) Implementation
(e) Testing and Evaluation
(f) Maintenance

Choose three sub-disciplines or tasks within these sub-disciplines that involve a mathematical approach, and illustrative them with examples.

(a) Requirement Analysis/Elicitation: The software engineering firm will send business analysts to meet with the customer and review their proposed application. Specific attention will be paid to the expected users of the application, the proposed use cases, and the functional and nonfunctional requirements of the expected system. The results of the discussion will be aggregated in a Requirement Specification document.

(d) Implementation: After the system has been designed, individual software engineers will implement each requirement in code. The implementation workflow is dependent on what model of software engineering is being followed. In Agile development, each feature will be developed during a sprint phase, with strong collaboration between developers and with customers. In a Waterfall method, the entire system will be implemented sequentially.

(e) Testing: After the system has been implemented, it is subjected to rigorous testing. Testing will focus on typical use cases as well as edge cases and system-breaking cases. Tests vary in scope, from unit testing on an individual component to interoperation testing on two connected components to system testing on the entire system.

4. **I.E Systems**

(a) _____ linked libraries can support shared library code, allowing one copy of a library routine to be used by several different processes.

absolute **relative** **static** **dynamic** **none of these is correct**

(b) When it is not known at compile time where a process will reside in memory, _____ code must be generated.

logical **physical** **absolute** **relocatable**

(c) A UNIX process calls *fork()* to create a child process as shown: *pid = fork()*;

i. What value will be assigned to pid in the parent process by the call to *fork()*?

the parent's process id **the child's process id** **zero none of these**

ii. What value will be assigned to pid in the child process by the call to *fork()*?

the parent's process id **the child's process id** **zero none of these**

(d) The Banker's algorithm is used for deadlock _____.

denial **prevention** **avoidance** **recovery**

(e) Belady's anomaly can affect the performance of the _____ page replacement algorithm.

FIFO **LRU** **optimal** **SJF**

(f) _____ access files are made of fixed length records that allow programs to read and write records in no particular order.

sequential **direct** **logical** **none of these is correct**

(g) When an I/O request is being handled for a user's process, which term refers to the policy of returning control to the user process before the I/O is completed?

synchronous I/O **asynchronous I/O** **delayed I/O** **none of these**

(h) Which multithreading model requires that a new kernel thread be created for each new user thread?

many-to-one **one-to-one** **many-to-many** **none of these is correct**

(i) A process that does not affect, and is not affected by, another process is referred to as:

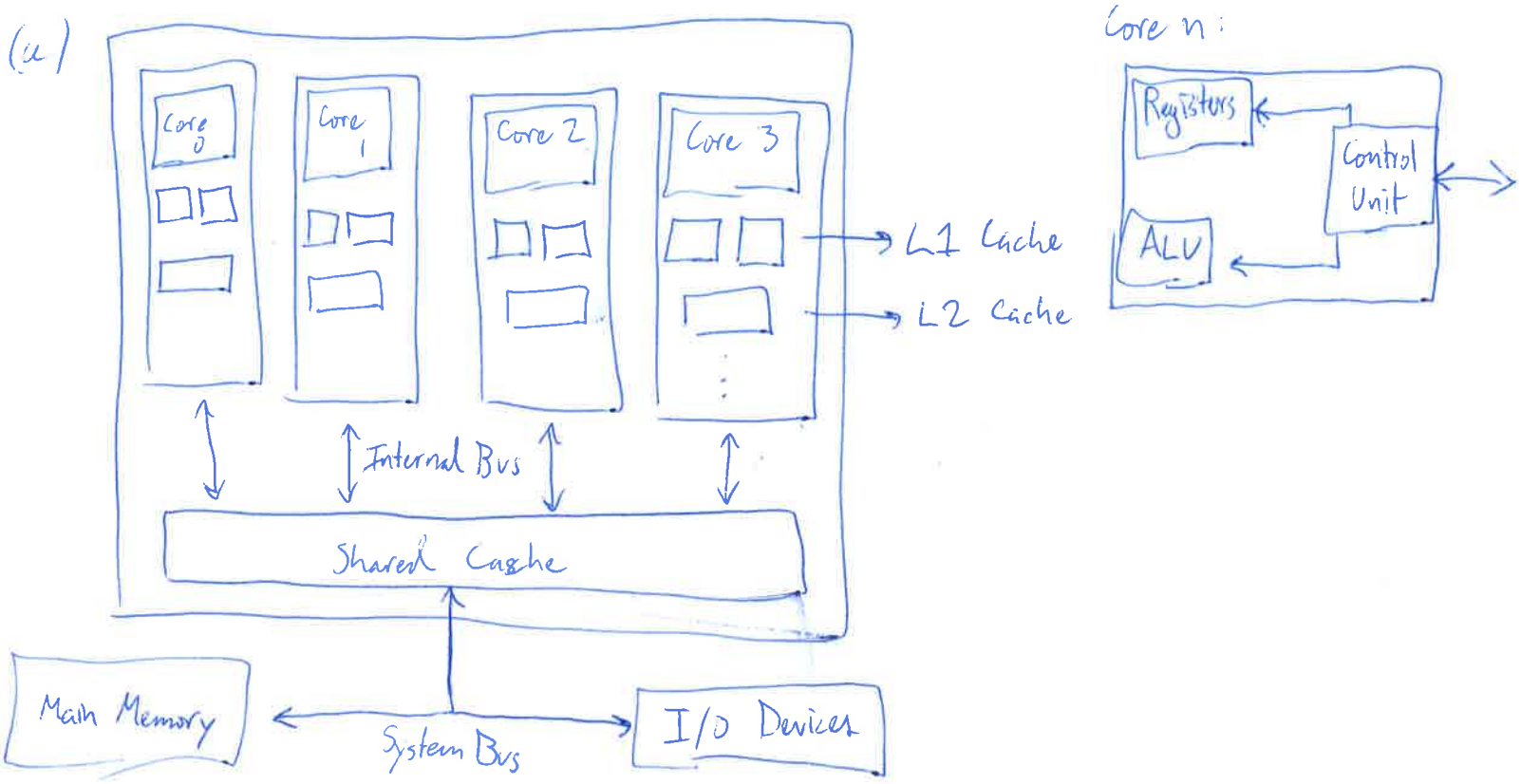
static **independent** **cooperating** **dynamic** **unbounded**

5. **II Computer Organization**

   (a) Draw an architecture of a quad-core processor and discuss the role of each module in your diagram.

   (b) Find a binary representation of the decimal number 0.1.

(a)



Each core contains:

Control Unit - Sends instructions to the ALU, data to the registers, and processes their outputs

Arithmetic Logic Unit - Implements logic and arithmetic functions in hardware

Registers - hold vital processing information such as calculations/I-O results, data accessed from memory, and the memory address of the program instructions

Each core has private cache for storing frequently used data locally. (The diagram contains 2 levels of private cache)

All cores are connected to a shared cache, which trades off access speed with ease of access for all cores, by an internal bus.

The entire processor is connected to main memory and I/O devices by the system bus.

(b) 0.1 in binary

$0.1 \times 2 = 0.2$
$0.2 \times 2 = 0.4$
$0.4 \times 2 = 0.8$
$0.8 \times 2 = 1.6$
$0.6 \times 2 = 1.2$
$0.2 \times 2 = 0.4$
$0.4 \times 2 = 0.8$
$0.8 \times 2 = 1.6$
$0.6 \times 2 = 1.2$
$0.2 \times 2 = 0.4$
⋮

$$0.1_{10} = 0.0001100110011_{2}$$

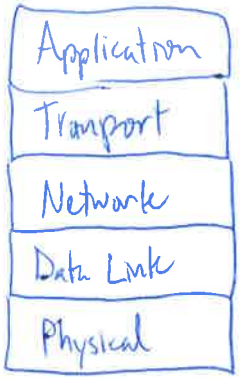
the repeating portion will be truncated at a varying length depending on the required precision.

6. **II.D Networking and Communications**

   (a) Draw a diagram showing layers of the Internet Protocol Stack and briefly discuss role of each layer.

   (b) Describe functions of each layer when a file is transferred from a source to destination using (*file transfer protocol* (FTP).

(a)



Application: establishes the connection between the client-side application and the server-side application. Issues requests for data and fulfills requests with data.

Transport: ensures the application-to-application delivery of the requested data. Data guarantees vary in the transport layer protocol used and include in-order transfer, reliable data transfer, error detection and re-transmission, and authentication.

Network: first layer implemented on end systems AND network hardware. Implements a routing algorithm to select the best path along which to forward the packet in processing.

Data Link: oversees the interface between two specific network devices. Handles the detection and correction of bit errors incurred by the transfer medium.

Physical: transfer medium; regulates the transmission of bits from one device to the pre-determined destination device.

(b) FTP

Application layer: The source and destination machines establish a client-server connection. The client machine issues a request for a file and the server machine begins transmitting the file.

Transport layer: The FTP protocol uses a TCP connection, so the two machines establish a connection through a 3-way handshake. The TCP ensures that the entire file is transferred reliably. This involves the detection of transmission errors and retransmission of missed packets. The packets are then reassembled in order, and delivered to the application layer.

Network layer: The network layer resolves a path from source to destination machine through the network. Typically, a forwarding table is pre-populated and refreshed at regular intervals so that a packet can be forwarded with maximum throughput.

Data Link layer: Each packet is overseen as it bounces from one network device to another. CRC is used to detect bit errors in the result and correct them or request retransmission.

Physical: Bits are transferred through a physical cable, wire, or wireless medium.

Networking Joke: Expect more UDP applications in the age of COVID, as developers are more hesitant to use TCP's three-way handshake. 😷 :)

7. **III.A Algorithms and complexity**

Describe an algorithm that takes two input lists of integers $A = a_1, ..., a_n$ and $B = b_1, ..., b_m$ and delivers the list of all the elements that belong to A but not to B. A and B do not contain redundant elements, however, the elements of A and B might have a large range.

The algorithm should run in $O(n \log m + m \log m)$ time.

```
def ListDifference(A, B):
    # a brute force approach will likely take O(nm), so B will be sorted to allow
      for faster search
    MergeSort(B)                    # O(mlogm)
    for i in [1...n]:               # O(n)
        BinarySearch(B, a_i)        # O(logm)
        if found:
            pass
        else:
            result.append(a_i)
    return result
```

The preceding algorithm first sorts B ($O(m \log m)$) then binary searches through B for each element in A. The result will be the list difference of A with B, and the algorithm will run with time $O(m \log m + n \log m)$.

8. **Automata and language theory**

Consider the following grammar:

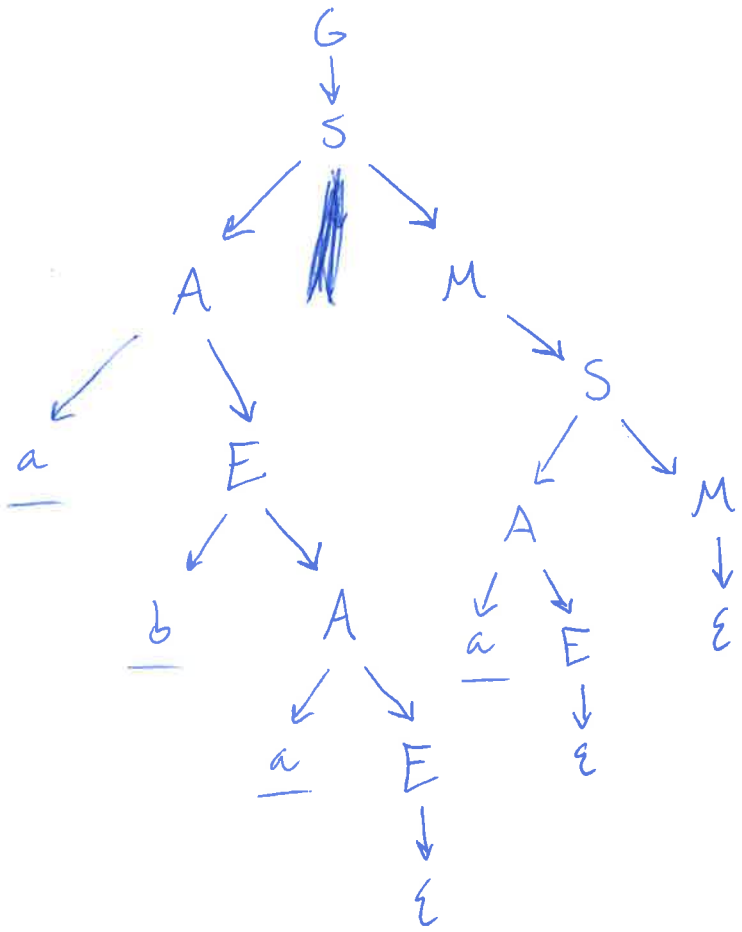
$G \longrightarrow S\ \$\$$
$S \longrightarrow A\ M$
$M \longrightarrow S \mid \epsilon$
$A \longrightarrow a\ E \mid b\ A\ A$
$E \longrightarrow a\ B \mid b\ A \mid \epsilon$
$B \longrightarrow b\ E \mid a\ B\ B$

(a) Describe the language that the grammar generates in English.

(b) Show a parse tree for the string `a b a a`.

(c) Is the grammar LL(1)? If so, show the parse table; if not, identify a prediction conflict.

(b)



(a) The grammar shown above generates the language on $\Sigma = \{a, b\}$ such that every string contains more a's than b's.

9. **III.C Discrete Structures**

Recall that the Hamiltonian Cycle Problem is the problem of deciding, on input graph G, whether G has a cycle that visits all the nodes exactly once. Show that this problem is polynomial time decidable if the input is restricted to the graphs with the property that each node has at most two neighbors (i.e., at most two adjacent nodes).

With input restricted to 2 neighbors per node, the Hamilton Cycle Problem becomes a variant of depth-first search.

1 Select a starting node V_0 and initialize a list of all nodes visited with V_0.

2 Traverse one of V_0's edges, place the new node V_i in the visited list. Since each node can only have two neighbors, there is only one more edge to follow for V_i.

3 Repeat the process of visiting a new node, placing it in visited, and following the other edge until the algorithm attempts to insert a duplicate node. That means a cycle has been found.

4 Check if visited is equal to the list of nodes in G. If so, a Hamilton Cycle has been found. If not, a Hamilton cycle cannot exist given the restrictions on the input.

This algorithm is decidable in polynomial time $O(V+E)$.
Additionally, if any node has fewer than 2 edges, a Hamilton cycle is impossible and the algorithm terminates.

10. **IV Other Topics**

Give a detailed explanation of any one approach to machine learning. Give a substantial example that illustrates the technical operation of the approach, and demonstrates interesting knowledge learned.

Naive Bayes

The supervised classification procedure Naive Bayes attempts to classify an example with feature vector $\vec{x} = [x_1, \ldots x_n]$ into class c_i by optimizing the conditional probability $P(c_i | \vec{x})$. To do so, the procedure uses Bayes Rule from statistics

$$P(c_i | \vec{x}) = \frac{P(c_i) \cdot P(\vec{x} | c_i)}{P(\vec{x})}.$$

$P(\vec{x} | c_i)$ cannot be calculated directly, so Naive Bayes uses a simplifying assumption that each feature in $\vec{x}$ is independent of all other features. This assumption is not statistically sound, but does yield favorable results in practice.

The primary use of Naive Bayes is in text classification tasks with limited domains such that a neural network is impractical. Take, for example, the task of determining whether a Tweet contains extremist/conspiratorial content. Given a training sample of Tweets (tokenized by word), we can compute the prior probabilities $P(x_1 | c_i)$, $P(x_2 | c_i)$, ... which equate to the probability that a Tweet containing word x_i is extremist. Then, using the simplifying assumption, $P(\vec{x} | c_i) = P(x_1 | c_i) \cdot P(x_2 | c_i) \ldots \cdot P(x_n | c_i)$. The calculation of $P(\text{extremist} | \vec{x})$ and $P(\text{not extremist} | \vec{x})$ is thus straightforward. Once each conditional probability is calculated, classify $\vec{x}$ in the class with the highest conditional probability.