Department of Computer Science
Core Graduate Comprehensive Exam
Fall 2020

- **Answer the questions on the paper supplied.**
- **Answer question 1 or 2. Answer question 3 or 4 or 5. Answer question 6 or 7 or 8 or 9 or 10. Answer question 11 or 12 or 13 or 14. You should answer a total of four questions. Please Note: If you answer more than one question in one group, only the one with the LOWEST score will be counted.**
- **Start each question on a new page. Write on only one side of the paper.**
- **Write your SIX-DIGIT Texas State ID in the top right corner of each page of your answer. Do NOT put your name anywhere on the answers.**
- **Put the number of the question being answered in the top left corner of each answer page. Put the CORRECT question number to avoid missing your answer.**
- **If the answer to a question is written on more than one page, number the pages consecutively.**

Group 1

1. **CS 5329 Algorithm Design and Analysis**
   {*from Dr. Hwang*}

   (a) Write the procedure QUICKSORT (A, p, r) and PARTITION(A,p,r).

   (b) PROVE and/or DERIVE the computation time T(n) for QUICKSORT (A, p, r) if the partition is always 30% and 70% split, where n is size of A, p & r are first index and last index of array A or of a sub array of A.

2. **CS 5329 Algorithm Design and Analysis**
   {*from Dr. Metsis*}

   Question: Sorting

   The pseudocode below shows an implementation of the Quicksort algorithm.

```
QUICKSORT(A, p, r)
  if p < r
      q = PARTITION(A, p, r)
      QUICKSORT(A, p, q − 1)
      QUICKSORT(A, q + 1, r)
```

```
PARTITION(A, p, r)
  x = A[r]
  i = p − 1
  for j = p to r − 1
      if A[j] ≤ x
          i = i + 1
          exchange A[i] with A[j]
  exchange A[i + 1] with A[r]
  return i + 1
```

a) Assume the following input array: A = [13, 19, 9, 5, 12, 8, 7, 4, 21, 2, 6, 11]. Show the state of the array A after the first call of the Partition algorithm.

b) What is the worst-case and what is the expected (average) computational complexity of the above algorithm? What type of input would case the worst-case complexity?

c) Modify the above algorithm in a way that it avoids the worst-case complexity with very high probability. Provide the pseudocode of your solution.

Group 2

3. **CS 5346 Advanced Artificial Intelligence**
   {*from Dr. Ali*}

   (a) Consider the Wumpus environment E as a 4 X 4 grid of rooms (E2,3 represents 2nd row and 3rd column). Sensors have detected the following facts:

   B2,1; B1,2; B2,3; B3,2 Where Bi,j indicates Breeze in the i-th row and j-th column. Breeze is not detected in any other room.

   Write rules in propositional logic to determine the position of the Pit and change them into Conjunctive Normal Form.

   Using resolution rule, develop a proof tree to determine the location of the Pit.

   (b) Represent the following story in propositional logic and then transform it to the conjunctive Normal Form:

   "James, Stevens and Nelson hold the jobs of dentist, pilot, and marine (not necessarily in that order). James owes the dentist $39. The marine's spouse prohibits borrowing money. Stevens is not married."

   Using resolution rule develop a proof tree and determine which person has which job.

4. **CS 5391 Survey of Software Engineering**
   {*from Dr. Chen*}

   Describe the similarities of the agile and the cleanroom software development approaches.

5. **CS 5391 Survey of Software Engineering**
   {*from Dr. Palacios*}

   A Software Process Improvement (SPI) framework includes, besides the Software Process being examined, the following elements:

   - Assessment,
   - Capability Determination, and
   - Improvement Strategy.

   (a) Define these three elements, and

   (b) Describe the overall improvement approach, providing an illustration of the SPI framework with a directed graph (with named edges) that combines these three elements and the Software Process.

Group 3

6. **CS 5306 Advanced Operating Systems**
   {*from Dr. Tamir*} Assume a single core system implementing an intra-core preemptive HRRN scheduling policy with a slice size of 1 second. Further assume that at time T there are 3 tasks {T0, T1, T2}in the Ready Queue of the core with no task in the execution slot of the core. Additionally, assume that the tasks are compute-bound with no I/O whatsoever. Let {P0, P1, P2} be the remaining execution time of {T0, T1, T2} respectively and let {P3, P4, P5} be the current wait time of {T0, T1, T2} respectively, where:

   P0 = 1
   P1 = 4
   P2 = 2
   P3 = 1
   P4 = 2
   P5 = 3

   1)Clearly describe the state of the system in each of the first 20 seconds following time T.

7. **CS 5306 Advanced Operating Systems**
   {*from Dr. Chen*}

   Design a distributed algorithm to synchronize physical clocks in a distributed system in a room.

8. **CS 5332 Data Base Theory and Design**
   {*from Dr. Hwang*}

   Draw an E/R Diagram for an ordering system design.

   The system must include at least the Order (Order number), the Product (Product number), and the Customer (Custom number). You must provide essential attributes for all relations you create. You must also normalize all relations up to 3NF or BCNF. You must specify PKs, FK's correctly. You must also specify all relationships. You must add additional relations and relationships to the design if it is required for a correct E/R model. The E/R diagram must use crow feet notation.

9. **CS 5332 Data Base Theory and Design**
   {*from Dr. Ngu*}

   Consider the following typed relational schema describing books and borrowers in a library:

   Book(<u>isbn:integer</u>, title:string, publisher:string, year:integer)
   Author(<u>isbn:integer</u>, <u>name:string</u>, rank:integer)
   Borrower(<u>bid:integer</u>, name:string, address:string)
   Borrowings(<u>isbn:integer</u>, <u>borrower:integer</u>, whenTaken:date, whenReturned:date)

   Assume the following:

   - the fields that are underlined are the primary keys of the relations
   - there is a referential integrity constraint between "borrower" in relation Borrowings and "bid" in relation Borrower
   - there are also referential integrity constraints between "isbn" in relation Book and "isbn" in relations Author and Borrowings
   - dates are represented as a count of the number of days since Jan 1 2020
   - there is a system function `today` that returns today's `date`
   - the relational operators $<$ and $>$ allow you test *before* and *after* on `dates`
   - all loans are for a period of 14 days
   - the `whenReturned` field is `null` until the book is returned
   - the `rank` field specifies whether the author is the first, second, ... author

   Implement solutions to the following:

   (a) *(4 marks)* Write an SQL query to show all the books (`title` only) written by a particular author called "Ullman".

   (b) *(4 marks)* Write an SQL query to show which books (`title` only) have multiple authors.

   (c) *(4 marks)* Write an SQL query to show which books (`title` only) have never been borrowed.

   (d) *(4 marks)* Write an SQL query to show which books (`title` only) are currently out on loan.

   (e) *(4 marks)* Write an SQL query to list overdue books (`title`) and the borrower's name

10. **CS 5310 Network and Communication Systems**
    {*from Dr. Peng*}

    (a) Describe the steps and protocols used when you send an email message using a computer in CS lab to a friend of yours who uses an email server on a campus in California.

    (b) Describe briefly why network layer is not an end-to-end layer.

Group 4

11. **CS 5338 Formal Languages**
{ *from Dr. Gao* }

    (a) Draw a PDA M that recognizes language $L(M) = \{b^n a^n : n \geq 0\}$.

    (b) Use operators = (meaning equal) and $<$ (meaning less than) to rank the following abstract machines in terms of computability.
    DTM (deterministic Turing Machine)
    NDTM (nondeterministic Turing Machine)
    DFSM (deterministic FSM)
    NDFSM (nondeterministic FSM)
    DPDA (deterministic PDA)
    NDPDA (nondeterministic PDA)

    (c) From the following abstract machines, underline the ones that must halt at all times.
    DFSM
    NDFSM
    NDFSM without ε-transitions
    DPDA
    NDPDA
    NDPDA without ε-transitions
    DTM
    NDTM

    (d) The P versus NP problem is a major unsolved problem in computer science. Assuming P $\neq$ NP, use a Venn diagram to show the logical relations for sets P, NP, and NP-complete.

12. **CS 5338 Formal Languages**
{ *from Dr. Singh* }

    What is Pumping Lemma for regular languages? Using Pumping Lemma prove $L = \{0^n 1 0^n | n \geq 1\}$ is not regular.

13. **CS 5318 Design of Programming Languages**
{ *from Dr. Shi* }

    (a) Algol-60 provided two parameter-passing mechanisms: call-by-value and call- by-name.
        i. Explain these mechanisms.
        ii. Justify or criticize the statement that "the former is expensive for arrays and the latter interacts badly with side effects."
        iii. What parameter-passing mechanism(s) do C and Java use, and how do such languages deal with an array being passed as a parameter?

14. **CS 5351 Parallel Processing**
{ *from Dr. Burtscher* }

Assume we want to parallelize the outer loop of the following code on a shared-memory system without caches.

```
for (int i = 0; i < n; i++) {
  int a = i;
  for (int j = 0; j < i; j++) {
    ... // do O(1) work here that reads variable a
    a += 3;
    ... // do some other O(1) work here that reads variable a
  }
}
```

1) Which of the following three workload partitioning schemes results in the least amount of load imbalance and why: blocked, cyclic, or block-cyclic?

2) If the system had a data cache, would a different workload partitioning scheme be better? If not, why not? If so, under what conditions and why?

3) Assuming that *n* is a large positive number and that each *i*-loop iteration is assigned to a separate thread (and no other parallelization is used), what is the parallel efficiency?

4) Assume each thread has a private variable called "*rank*" that contains the thread's unique rank. Assume further that there is a shared variable called "*threads*" that contains the number of running threads and that all ranks are in the range 0 through *threads* - 1. Using these variables, rewrite the above code such that it assigns the *i*-loop iterations cyclically to the threads.

5) Rewrite the above code such that the loop-carried data dependence on variable "*a*" is eliminated but the code maintains its semantics.