2012 Fall BIOSTAT 615/815 Midterm

Tuesday October 23rd, 08:40AM-10:00AM
UMID : ( )
UNIQNAME : ( )
NAME : ( )

Problem 1. True/False Questions (10 pts - 2pts each)

Write down **T** (True) or **F** (False) for each of the statement below.

1. ( ) Divide-and-conquer algorithms can only be implemented by recursions.
2. ( ) The time complexity of the Hanoi tower problem can be reduced from exponential to polynomial time complexity using dynamic programming.
3. ( ) The time complexity of `insert` algorithm in a singly linked list is $\Theta(n)$.
4. ( ) The worst-case time complexity of Quicksort is $\Theta(n^2)$.
5. ( ) The time complexity of forward-backward algorithm is the same to the time complexity of Viterbi algorithm.

Problem 2. Short answer questions (10pts - 5pts each)

Write down the expected output for each program.

(a)

```
#include <iostream>  // prog2a.cpp
int main(int argc, char **argv) {
  int a[5] = {515, 615, 715, 815, 915};
  int& b = *(a+1);
  int* c = &a[1];
  ++b;
  ++c;
  std::cout << a[1] + c[1] << std::endl;
  return 0;
}
```

(b)

```
#include <iostream>

int foo(int n) {
  if ( n == 1 ) { return 1; }
  else { return foo(n/2)+1; }
}

int main(int argc, char** argv) {
  std::cout << foo(8) << std::endl;
  std::cout << foo(10) << std::endl;
  return 0;
}
```

Problem 3 - Long answer questions (20pts)

(a) (10pts) The following piece of code is supposed to compute the average across all the input arguments (except for the program name itself) and print out the average. But it won't compile and run correctly due to one or more errors. Describe how to correct it to make it work. Assume that all the input arguments are integers and given correctly.

```
#include <iostream>
#include <cstdlib>

int main(int argc, char** argv) {
  int sum = 0;
  for(int i=0; i < argc; ++i) {
    sum += argv[i];
  }
  std::cout << "Average = " << sum/argc << std::endl;
  return 0;
}
```

(b) (10pts) Below is an C++ function that can be called in R to compute square of each element in a vector. But it is missing one critical line, so it will throw an error. Add one line in a proper location to fix the problem.

```
#include <R.h>
#include <Rinternals.h>
#include <Rdefines.h>

SEXP square(SEXP in) {
  SEXP out;
  int nr = length(in);
  PROTECT( out = allocVector(REALSXP, nr) );
  PROTECT(in = AS_NUMERIC(in));
  double* p_in = NUMERIC_POINTER(in);
  double* p_out = REAL(out);
  for(int i=0; i < nr; ++i) {
    p_out[i] = p_in[i]*p_in[i];
  }
  return (out);
}
```

Problem 3 - Quicksort (10pts)

Describe the key idea of QuickSort algorithm, by dividing the algorithm into three parts. (a) Divide, (b) Conquer, and (c) Combine. Briefly explain why the algorithm is expected to correctly sort the input array. You don't have to provide the full pseudo-code, but you may if you want.

Problem 4 - Factorial (10pts)

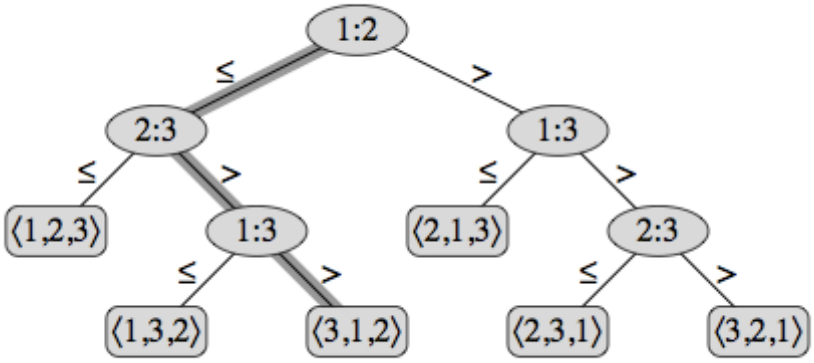
The following function `fac` calculates the factorial of an integer, but is has a serious limitation. Describe what the limitation is, and explain how to modify the function to overcome the limitation.

```
int fac(int n) {
  int ret;
  for(ret=1; n > 0; --n) { ret *= n; }
  return ret;
}
```

Problem 5 - Decision Tree for Comparison-based Sort (10pts)

Below is the InsertionSort algorithm, and the image below is the binary decision tree representing the full behavior of the algorithm for input of three elements. Draw a similar decision tree for 4 input elements.

Data: An unsorted list $A[1 \cdots n]$
Result: The list $A[1 \cdots n]$ is sorted
for $j = 2$ **to** n **do**
  $key = A[j]$;
  $i = j - 1$;
  while $i > 0$ *and* $A[i] > key$ **do**
    $A[i + 1] = A[i]$;
    $i = i - 1$;
  end
  $A[i + 1] = key$;
end



Problem 6 - Hidden Markov Model (20 pts)

Consider the following 2-states, 3 observation hidden Markov model with $\lambda = (\pi, A, B)$, where π is initial state distribution, with the following set of parameters.

$$\pi = \begin{pmatrix} 0.8 \\ 0.2 \end{pmatrix}, \qquad A = \begin{pmatrix} 0.8 & 0.2 \\ 0.4 & 0.6 \end{pmatrix}, \qquad B = \begin{pmatrix} 0.1 & 0.4 & 0.5 \\ 0.7 & 0.2 & 0.1 \end{pmatrix}$$

(a) When the observation is (3,1) for the first two days, what is the most likely sequence of states in these two days? Explain your answer with intermediate results

(b) Using forward-backward algorithms, compute the $\Pr(q_t|\mathbf{o}, \lambda)$ for day 1 and day 2. If the calculation is too complicated, you may leave the equation as is. You may want to utilize the equation below to recall forward-backward algorithm.

$$\begin{aligned} \alpha_t(i) &= \sum_{j=1}^{n} \alpha_{t-1}(j) a_{ji} b_i(o_t) \\ \alpha_1(i) &= \pi_i b_i(o_1) \\ \beta_t(i) &= \sum_{j=1}^{n} \beta_{t+1}(j) a_{ij} b_j(o_{t+1}) \\ \beta_T(i) &= 1 \end{aligned}$$

Problem 7 - Dynamic Programming (10pts)

Calculate the minimum edit distance between two words `MAN` and `AND`. Draw the matrix of dynamic programming storage containing the optimal cost to each node, and mark optimal move as to reach to each node. If there are multiple optimal moves, you may arbitrarily choose the optimal move.