CS 315 Data Structures Spring 2008

Lab # 3, February 12, 2008

This lab will weigh *3 points*.

In this lab, we will accomplish two tasks:

1. We will continue the discussion of Project # 1 which is due in two weeks.
2. We will work on some problems related to Chapter 2.

The goal of the second part of the lab is to analyze and estimate the number of operations performed by some simple iterative programs.

First you should review the summation formula for arithmetic series.

Arithmetic series: If $A_1 + A_2 + \ldots + A_n$ forms an arithmetic series (i.e., the difference between consecutive terms is a constant), then:

$A_1 + A_2 + \ldots + A_n = n\ A$ where A = the average of the first and the last terms, i.e., $A = (A_1 + A_n)/2$

O notation: O notation is used to approximate a function in terms of simple functions. These approximations hold when n is large. The simple functions are n^k, log n and c^n. Most of the estimates will be one of the terms or some product of these terms such as n log n. Basic rules for simplification are as follows:

1) $c^n >> n^k >>$ log n >> d (for any c > 1 and any k > 0, and for any d) Here, c, k, d etc. are constants.

2) $n^K >> n^L$ if K > L

3) approximate A + B by A if A >> B. So a sum involving many terms can be approximated by the fastest growing term.

4) Finally in a term like $5\ n^3$, you can drop a constant coefficient and approximate it as n^3. The final resulting expression gives an order of magnitude estimate.

We use O notation to express this fact.

Example: Consider the expression (3n + 5 log n + 6) (2 log n + 23) + 4n. We will check that this expression is O(n log n).

The sum 3n + 5 log n + 6 = O(n) since n is the biggest term
2 log n + 23 = O(log n) so the product is O(n log n).

Analysis of iterative algorithms

A systematic approach to the analysis is as follows: A straight-line program is a program that has no loops. To analyze a straight-line program, compute the time complexity of each instruction and add them.

Analysis of loops: Start with the innermost loop, and calculate the number of steps inside out. In the lecture, we presented two examples – the analysis of insertion sorting and selection sorting. Next we will look at the maximum subsequence sum problem that is discussed in the text. (Handout from the text will be given in the lab.)

Problems:

For each of the following algorithms, determine the number of operations of the specified type. You are to answer these questions without using the computer.

1)
```
for (int j = 1; j < 100; j++) {
    B[j] = 0;
    for (int k = j+ 1; k < 100; ++k)
      B[j] += B[k];
  }
```

How many times is the instruction **`B[j] += B[k]`** executed?

2)
```
for (int j = 0; j < 256; ++j)
    for (int k = 0; k < 256; ++k)
      A[j][k] = B[j][k] + C[j][k];
```

How many times is the instruction **`A[j][k] = B[j][k] + C[j][k]`** performed?

3)
```
A[0] = 1; A[1] = 1;
  for (int j = 2; j < 1000; ++j)
    if (j % 2 == 0)
       A[j] = A[j-1]+3*A[j-2];
```

```
    else
       A[j] = A[j-1]+A[j-2];
```

What is the total number of operations performed by the above program?

For the following problems, first determine the number of operations of the specified type as a function of n, the input size. Then express your answer using O notation.

4) Shown below is a function that takes as input a vector of n integers and determines the maximum and the minimum numbers in the array. Determine the number of comparisons (between keys) performed by the program as a function of n in the best-case and in the worst-case. In the code, swap(x,y) swaps the keys x and y. (Assume that n is even.)

```
void findminmax (vector<int> A, int& min, int& max) {
  int n = A.size();
  for (int j = 0; j < n-1; j = j+2)
     if (A[j] > A[j+1]) swap(A[j], A[j+1]);
  min = A[0]; max = A[1];
  for (int j = 0; j < n-1; j = j+2) {
    if (A[j] < min) min = A[j];
    if (A[j+1] > max) max = A[j+1];
  }
}
```

5) The following is an inefficient way to determine if a given integer N is prime. Determine the number of divisions performed by this function in the best-case and the worst-case.

```
bool isPrime (int N) {
 if (N == 2) return true;
 if (N % 2 == 0) return false;
 for (int j = 3; j < N/2; j = j + 2)
   if (N % j == 0) return false;
 return true;
}
```

What should be submitted and when?

The solutions to the above problems are due at the end of the lab today. You are to write the answers to these problems on the given sheet and submit before you leave.