22C:21 Lecture Notes

Jan 25th, 2006

Example 3.

```
for(i = 0; i < n; i++)
      for(j = 0; j < i; j++)
              System.out.println("Hello");
```

Here n is the input size. We want to estimate the running time of the above code as a function of n. Just as in Example 2, the above code can be expanded to

```
1.  i = 0
2.  if i >= n then goto line 10
3.  j = 0
4.  if j >= i then goto line 8
5.  Output "Hello"
6.  j++
7.  goto 4
8   i++
9   goto 2
10
```

Note that the only difference between the above code and the code that we got by expanding Example 2 is in Line 4, where we now have the condition $j \geq i$ rather than $j \geq n$.

As in the earlier example, each of the above 9 lines of code will run in time that does not depend on n. So assume that on some hypothetical machine, Line i takes c_i units of time, for $i = 1, 2, \ldots, 9$. Let us now figure out how many times each line executes. Clearly, Line 1 executes once. Line 2 executes once for each value of $i = 0, 1, 2, \ldots, n$, for a total of $(n + 1)$ times. For values of $i = 0, 1, \ldots, n - 1$, the condition in Line 2 evaluates to false and we just drop down to Line 3. Therefore, Line 3 executes n times, once for each value of $i = 0, 1, 2, \ldots, n - 1$. When $i = 0$, Line 4 is executed once. When $i = 1$, Line 4 is executed twice, just when of $j = 0, 1$. When $i = 2$, Line 4 is executed three times, just when of $j = 0, 1, 2$. In general, for any value of i, $0 \leq i \leq n-1$, Line 4 is executed exactly $i+1$ times. Therefore, the total number of executions of Line 4 is

$$1 + 2 + 3 + \ldots + n = \frac{n(n+1)}{2}.$$

We will see later why the sum of the first n natural numbers is $n(n + 1)/2$. The calculation of how many times Line 5 is executed is very similar. For each value of i, Line 5 is executed i times, once for each value of $j = 0, 1, \ldots, i - 1$. Therefore, the total number of executions of Line 4 is

$$1 + 2 + 3 + \ldots + (n - 1) = \frac{(n-1)n}{2}.$$

Lines 6 and 7 are executed exactly as many times as Line 5, that is, $(n - 1)n/2$ times. Line 8 is outside the inner loop and is executed once for each value of $i = 0, 1, \ldots, n - 1$ for a total of n times. Similarly, Line 9 is exeecuted n times.

The following table summarizes our calculations.

Line 1	1 time
Line 2	$(n + 1)$ times
Line 3	n times
Line 4	$n(n + 1)/2$ times
Line 5	$(n - 1)n/2$ times

Line 6	$(n-1)n/2$ times
Line 7	$(n-1)n/2$ times
Line 8	n times
Line 9	n times

The total running time of Line i for $i = 1, 2, \ldots, 9$ is c_i times the number of times Line i is executed. The total running time of the code fragment is the sum of the total running time of each line. We get that the total running time of the code fragment is:

$$c_1+c_2\cdot(n+1)+c_3\cdot n+c_4\cdot n(n+1)/2+c_5\cdot(n-1)n/2+c_6\cdot(n-1)n/2+c_7\cdot(n-1)n/2+c_8\cdot n+c_9\cdot n.$$

We separate out terms that contain n^2, terms that contain n, and terms that are just constants, to get the following expression:

$$n^2\cdot(c_4/2+c_5/2+c_6/2+c_7/2)+n\cdot(c_2+c_3+c_4-c_5/2-c_6/2-c_7/2+c_8+c_9)+(c_1+c_2).$$

This can be written as An^2+Bn+C, where A, B, and C are constants independent of n. Thus the running of this code fragment is also quadratic in n.

Arithmetic series. To see that $1+2+\cdots+n = n(n+1)/2$ let us denote the series $1+2+\cdots+n$ by S. Then,

$$\begin{aligned} S &= 1+2+\cdots+(n-1)+n \\ S &= n+(n-1)+\cdots+2+1 \\ 2S &= (n+1)+(n+1)+\cdots+(n+1)+(n+1) \end{aligned}$$

The second line above is obtained just by writing the right hand side from the first line in reverse order. The last line is obtained by adding up the corresponding terms in the first two lines. Since there are n terms on the right hand side of the third line, we get that $S = n(n+1)/2$. Therefore, the sum of the first n natural numbers is $n(n+1)/2$. To get the sum of the first $n-1$ natural numbers, simply replace n by $n-1$ and we get that this sum is $(n-1)n/2$.
Here is another example. Let n be a multiple of 10. What is the sum

$$1+2+\ldots+n/10?$$

Simply replace n by $n/10$ in the expression for the sum of the first n natural numbers and we get $n(n+10)/100$.

These summations are all examples of *arithmetic series*. The general form of an arithmetic series is:

$$a+(a+d)+(a+2d)+(a+3d)+\cdots+(a+(n-1)d).$$

Here a is the initial term and each subsequent term is obtained from the previous term by adding a value d. Using the expression for the sum of the first n natural numbers, we can get an expression for arithmetic series also, as follows.

$$\begin{aligned} a+(a+d)+(a+2d)+(a+3d)+\cdots+(a+(n-1)d) &= n\cdot a+d\cdot(1+2+3+\cdots+(n-1)) \\ &= n\cdot a+d\cdot\frac{(n-1)n}{2} \\ &= \frac{n}{2}\cdot(2a+(n-1)\cdot d) \\ &= n\cdot\frac{(a+(a+(n-1)\cdot d))}{2} \end{aligned}$$

The way to read this expression is that it is the number of terms (n) times the mean of the first and last terms.
Here are a couple of examples that illustrate the use of this summation.

Example 1. What is $5 + 15 + 25 + \cdots$ up to n terms? Here $a = 5$, $d = 10$ and therefore the answer is

$$\frac{n(5 + (5 + (n-1)10))}{2} = 5n^2.$$

Example 2. What is the number of lines of output printed when the following piece of code is executed? Assume that n is a multiple of n.

```
for(i = 0; i < n; i = i + 10)
        for(j = 8; j < i; j++)
               System.out.println("Hello");
```

When $i = 0$, no output is produced. When $i = 10$, two lines of output is produced. When $i = 20$, 12 lines of output is produced. When $i = 30$, 22 lines of output is produced. The final value of i for which the inner loop if executed, is $n - 10$. So we need to compute the sum $2 + 12 + 22 + \cdots$. The number of terms in this sum is $n/10 - 1$, $a = 2$, and $d = 10$. Using the formula for arithmetic series, we see that the sum is

$$\frac{(n/10 - 1)(2 + 2 + (n/10 - 2)10)}{2} = \frac{(n-10)(2 + 2 + (n-20))}{20} = \frac{(n-10)(n-16)}{20}.$$
