Use the technique shown in class for finding lower and upper bounds for a sum to prove that:

$$\sum_{i=0}^{n} i^5 \quad \in \quad \theta(n^6)$$

Announcements

Start work on Assignment #2 early so you can ask for help if you need it.

Assignment 2 Part A: Due Fri. Oct. 4.
Assignment 2 Part B: Due Tues. Oct. 8.

Office hours this week:

F 12:30, 1:30

WECS: CSC Revealed Clinic today at 2:30 in ECS 348. Volunteers welcome.
WECS has an e-mail listserv: sign up?
http://wecs.csc.uvic.ca/

Quicksort for assignment 2B:

1. pivot = first big integer value on the list.
2. Create 3 new lists, $List_1$, $List_2$ and $List_3$.
3. Traverse the list removing each cell and placing it at the end of the appropriate list:
   - $List_1$: for values less than the pivot.
   - $List_2$: for values equal to the pivot.
   - $List_3$: for values greater than the pivot.
4. Sort $List_1$ and $List_3$ recursively.
5. The answer is $List_1 \circ List_2 \circ List_3$.

```
public void maxSort(int size)
{     int i, t, maxPos;

      if (size <= 1) return;
      maxPos=0;
      for (i=1; i < size; i++)
            if (A[i] >= A[maxPos]  ) maxPos=i;

      t= A[maxPos];
      A[maxPos]= A[size-1];
      A[size-1] = t;

      maxSort(size-1);
  }
```

Prove by induction:

maxSort does n(n-1)/2 key comparisons on a problem of size n.

Critique this proof that n(n-1)/2 is correct:

Basis (n=0): 0*(0-1)/2=0 so the base case holds.

[Induction step]
Assume 1+2+ … + (n-1)= n(n-1)/2.
We want to prove that
1+2+ … + n= (n+1)n/2.

1+2+ … + n = [1 + 2 + … + n-1] + n =
n(n-1)/2 + n= (n+1)n/2 as required.

If it is a valid proof, how do you distinguish it from this "proof" that n(n+1)/2 is correct?

Basis (n=0): 0*(0+1)/2=0 so the base case holds.

[Induction step]
Assume 1+2+ … + n= n(n+1)/2.
We want to prove that
1+2+ … + n+ n+1= (n+1)(n+2)/2.

1+2+ … + n + (n+1) = [1 + 2 + … + n] + (n+1) = n(n+1)/2 + n+1= (n+1)(n+2)/2 as required.

If you are trying to prove something about a program (in this case the maxSort method), it is critical to discuss the operation of that program in your proof.

Don't be afraid of putting words/explanation into your proofs.

Induction proofs should be more than just some algebra.

Explain what you are doing at each step and indicate where you apply the induction hypothesis.

[Basis]

On a problem of size 1, the maxSort starts with this statement:

```
if (size <= 1) return;
```

and since size=1 for a problem of size 1, the maxSort returns without making any key comparisons.

The formula gives n(n-1)/2 which for n=1 is equal to 1 (1-1) / 2 = 0 as required.

[Induction step]
Assume that maxSort does n(n-1)/2 key comparisons on a problem of size n.
We want to prove that maxSort does (n+1)((n+1)-1)/2 = (n+1)n/2= $(n^2 + n)/2$ key comparisons on a problem of size n+1.

The initial call to maxSort for a problem of size n+1 has size=n+1.
Each iteration of the for loop has 1 key comparison:

```
for (i=1; i < size; i++)
        if (A[i] >= A[maxPos]  ) maxPos=i;
```

The iterations of this loop have i=1, 2, …, size-1 and hence the number of iterations is size-1=n. Thus, the initial call has n key comparisons at the top level of recursion. The problem remaining after the maximum key is exchanged with the one in position n has size n. By induction, this problem will involve n(n-1)/2 key comparisons arising from the call:
maxSort(size-1);

So the total number of key comparisons is:

n + $n(n-1)/2$

(Top level) (Solving problem of size n)

$= 2n/2 + (n^2 - n)/2 = (n^2 + n)/2$ as required.

How could we bound the key comparisons so that we know the number is in $\theta(n^2)$ without computing the exact number of them?

Or how could we bound the total amount of work done?

This tactic of getting lower and upper bounds on the work done by an algorithm can be very valuable for more complicated algorithms that are harder to analyze.