CS361 Homework #1 Solutions

1. According to Moore's Law (no relation), computers get twice as fast every 18 months. Suppose that I can currently solve a problem of size n within one week. What size of problem will I be able to solve in one week with a computer 3 years from now if the running time of my algorithm grows as:

   (a) $f(n) = \Theta(n)$? $4n$ (if the speed doubles in 18 months, it quadruples in 3 years.)
   (b) $f(n) = \Theta(n^2)$? $2n$
   (c) $f(n) = \Theta(\sqrt{n})$? $16n$
   (d) $f(n) = \Theta(2^n)$? $n + 2$
   (e) $f(n) = \Theta(4^n)$? $n + 1$
   (f) $f(n) = \Theta(2^{n/2})$? $n + 4$
   (g) $f(n) = \Theta(\log_2 n)$? n^4
   (h) $f(n) = \Theta(\log_{10} n)$? n^4

2. When we say $f(n) = O(\log n)$, why don't we need to state the base of the logarithm? Because changing the base only changes the log by a constant factor: $\log_b n = (\log n)/(\log b)$.

3. Is $2^{O(n)}$ the same as $O(2^n)$? Why or why not? No: changing the exponent by a constant can change the function by an exponential factor. For instance, $2^{2n} = 4^n = \omega(2^n)$.

4. There are two ways to represent a graph: an *adjacency matrix* where for every pair of vertices u, v there is a bit that tells you whether u and v are connected, and an *edge list* which gives a list of the edges (u, v). If a graph has N vertices and M edges, how many bits does it take to give the adjacency matrix, and how many bits does it take to give the edge list? Answer both in the *sparse case* where $M = \Theta(N)$, and in the *dense case* where $M = \Theta(N^2)$. Assume that each vertex is represented by a number between 1 and N (how many bits does each such number take?) and give your answers in terms of Θ.

   Answer. The adjacency matrix takes N^2 bits. For the edge list, since it takes $\log_2 N$ bits to represent a number from 1 to N and each edge consists of two vertices, it takes $2M \log_2 N = \Theta(M \log N)$ bits. In the sparse case where $M = \Theta(N)$ this is only $\Theta(N \log N)$, much more efficient than the adjacency matrix; in the dense case where $M = \Theta(N^2)$ this is $\Theta(N^2 \log N)$, so the adjacency matrix is more efficient.

5. For each pair of functions, state whether their relationship is $f = o(g)$, $f = \Theta(g)$, or $f = \omega(g)$:

   (a) $f(n) = \log \sqrt{n}$, $g(n) = \log(n^2)$: $f = \Theta(g)$.
   (b) $f(n) = 3^{n/2}$, $g(n) = 2^n$: $f = o(g)$ since $3^{1/2} = \sqrt{3} < 2$.
   (c) $f(n) = 2^n$, $g(n) = n^{\log n}$: $f = \omega(g)$. Take logs: $n = \omega(\log^2 n)$.
   (d) $f(n) = 2^{n+10}$, $g(n) = 2^n$: $f = \Theta(g)$ since $f/g = 2^{10}$ is a constant.
   (e) $f(n) = 2^{n+\log n}$, $g(n) = 2^n$: $f = \omega(g)$ since $f/g = 2^{\log n} = n$.

(f) $f(n) = n^n$, $g(n) = n!$: $f = \omega(g)$ since

$$\frac{f(n)}{g(n)} = \frac{n}{n-1}\frac{n}{n-2}\frac{n}{n-3}\cdots\frac{n}{1} \to \infty$$

6. Substitute a solution of the form $f(n) = A \cdot 4^n + B$ into the recurrence $f(n) = 4f(n-1) + 4$ and the base case $f(0) = 0$ and solve for A and B.

   The recurrence gives

$$\begin{aligned} f(n) &= 4f(n-1) + 4 \\ \Rightarrow 4^n A + B &= 4(4^{n-1}A + B) + 4 \\ &= 4^n A + 4B + 4 \\ \Rightarrow B &= 4B + 4 \\ \Rightarrow -3B &= 4 \\ \Rightarrow B &= -4/3 \end{aligned}$$

   and then the base case gives

$$\begin{aligned} f(0) &= 0 \\ A4^0 + B = A + B &= 0 \\ \Rightarrow A &= 4/3 \end{aligned}$$

   so the exact solution is

$$f(n) = (4/3) \times 4^n - 4/3 = (4/3)(4^n - 1) \ .$$

7. Substitute a solution of the form $f(n) = r^n$ into the recurrence $f(n) = f(n-1) + 6f(n-2)$ and solve for r. Hint: divide both sides by r^{n-2}. There may be more than one solution; which one matters when n is large?

$$\begin{aligned} f(n) &= f(n-1) + 6f(n-2) \\ \Rightarrow r^n &= r^{n-1} + 6r^{n-2} \\ \Rightarrow r^2 &= r + 6 \\ \Rightarrow r^2 - r - 6 &= 0 \end{aligned}$$

   Since $r^2 - r - 6 = (r-3)(r+2)$, the two roots are 3 and -2. As n grows, $(-2)^n$ oscillates and grows exponentially, but 3^n grows even more quickly, so $f(n) = \Theta(3^n)$.

8. Substitute a solution of the form $f(n) = An$ into the recurrence $f(n) = 2f(n/3) + n$ and solve for A. Substitute your final solution back in to the recurrence to make sure it works.

$$\begin{aligned} An &= 2An/3 + n \\ \Rightarrow A &= 2A/3 + 1 \\ \Rightarrow A/3 &= 1 \\ \Rightarrow A &= 3 \end{aligned}$$

   so $f(n) = 3n$.

9. Substitute a solution of the form $f(n) = n^\alpha$ into the recurrence $f(n) = 8f(n/4)$ and solve for α. Substitute your final solution back in to the recurrence to make sure it works.

$$\begin{aligned} f(n) &= 8f(n/4) \\ \Rightarrow n^\alpha &= 8(n/4)^\alpha \\ &= n^\alpha \times 8/4^\alpha \\ \Rightarrow 4^\alpha &= 8 \\ \Rightarrow \alpha &= \log_4 8 = 3/2 \end{aligned}$$

so $f(n) = n^{3/2}$.

10. Given the base case $f(0) = 1$ and the recurrence $f(n) = 3f(n-1) + n$, calculate $f(n)$ for the first few values of n, up to $n = 8$ or 9. Find the pattern in these numbers and propose a form for $f(n)$.

    This is somewhat challenging: if you think $f(n)$ grows roughly as fast as some simple function $g(n)$, you might want to check the ratios $f(n)/g(n)$ to figure out the multiplying constant. Then, see if adding some correction term gives you $f(n)$.

    Finally, substitute your final answer back into the recurrence and base case and confirm that it works.

    Answer. Here are the first few values:

n	0	1	2	3	4	5	6	7	8
$f(n)$	1	4	14	45	139	422	1272	3823	11477

    From the recurrence, we might guess that when n is large, the driving term n is unimportant compared to the exponential growth caused by the homogeneous term $3f(n-1)$, and so that $f(n) = \Theta(3^n)$. Indeed, if we look at the ratios between $f(n)$ and $f(n-1)$, these converge to 3.

    Inspired by this, we divide $f(n)$ by 3^n, and find that the result converges to $7/4$. We conclude that $f(n)$ is $(7/4)3^n$ plus or minus a small correction.

    To find this correction, we subtract $(7/4)3^n$ from $f(n)$ and get

n	0	1	2	3	4	5	6	7	8
$f(n)$	1	4	14	45	139	422	1272	3823	11477
$f(n) - (3/4)3^n$	$-3/4$	$-5/4$	$-7/4$	$-9/4$	$-11/4$	$-13/4$	$-15/4$	$-17/4$	$-19/4$

    This correction is $-n/2 - 3/4$, so the final answer is $f(n) = (7/4)3^n - n/2 - 3/4$.

    Finally, we check the base case $f(0) = 7/4 - 0 - 3/4 = 1$, and the recurrence:

$$\begin{aligned} f(n) &= 3f(n-1) + n \\ (7/4)3^n - n/2 - 3/4 &= 3((7/4)3^{n-1} - (n-1)/2 - 3/4) + n \\ &= (7/4)3^n - 3(n-1)/2 - 9/4 + n \\ &= (7/4)3^n - n/2 + 3/2 - 9/4 \end{aligned}$$

    and it works.

11. In the following program, the global variable `tick` represents the running time. Let $f(n)$ be the total number of `ticks` for calling `calc(n)`. Find the recurrence relation for $f(n)$ and its base cases. Explain which part of the recurrence is the homogeneous term, and which is the driving term. Finally, solve this recurrence and determine $f(n)$ within Θ.

```
calc(int n) {
  if (n <= 0) return;
  else {
    calc(n-1);
    for (int i = 0; i < n; i++)
      for (int j = 0; j < i; j++)
        tick++;
    calc(n-2);
  }
}
```

Answer. $f(n) = f(n-1)+f(n-2)+n(n-1)/2$. The homogeneous part $f(n) = f(n-1)+f(n-2)$ comes from `calc` calling itself twice on $n-1$ and once on $n-2$, and the driving term $0+1+2+\cdots+(n-1) = n(n-1)/2$ comes from the two nested `for` loops.

The solution to the homogeneous part is just like the Fibonacci numbers, $f(n) = \varphi^n$ where φ is the golden ratio $(\sqrt{5}+1)/2 \approx 1.618$. Since the driving term is only polynomial, it is insignificant compared to this exponential growth, and $f(n) = \Theta(\varphi^n)$.

12. A *quasipolynomial* is a function of the form $f(n) = 2^{\Theta(\log^k n)}$ for some constant $k > 0$ (where $\log^k n$ means $(\log n)^k$).

    (a) Show that a quasipolynomial function with $k > 1$ is ω of any polynomial and o of any exponential.

    (b) Show that if $f(n)$ and $g(n)$ are both quasipolynomial, then so is their composition $f(g(n))$.

    Answer. We have $\log f(n) = \Theta(\log^k n)$, while if $g(n)$ is a polynomial n^c, then $\log g(n) = c \log n = \Theta(\log n)$. If $k > 1$ we then have $\log f = \omega(\log g)$, which implies $f = \omega(g)$.

    On the other hand, if $g(n)$ is an exponential function 2^{cn}, then $\log g(n) = \Theta(n)$, and now $\log f = o(\log g)$.

    For the second question, suppose that $f(n) = 2^{\log^k n}$ and $g(n) = 2^{\log^\ell n}$. Then

$$f(g(n)) = 2^{\log^k 2^{\log^\ell n}} = 2^{(\log^\ell n)^k} = 2^{\log^{k\ell} n} \quad .$$

    (The same argument works if you Θ in the exponent with any constant.)

13. The research lab of Prof. Flush is well-funded, and they regularly upgrade their equipment. Brilliant Pebble, a graduate student, has to run a rather large simulation. Given that the speed of her computer doubles every two years, if the running time of this simulation exceeds a certain T, she will actually graduate earlier if she waits for the next upgrade to start her program. What is T?

    Answer. Suppose her simulation would take 4 years using her current computer. If she waits two years, her computer will be twice as fast, and her simulation will only take 2 more years after that, for a total of 4. This is the break-even point; if the simulation would take more than 4 years, she should wait!