Worksheet Fibonacci - Part V

Oliver Ray

Week 9

This worksheet concludes the Fibonacci study. You will analyse the time complexity of 3 functions from previous worksheets (written below with `double`'s instead of `int`'s and using a slightly more compact syntax) and you will compare how efficiently they can compute large Fibonacci numbers $f(n)$ by exploiting the so-called 'Big-Oh' notation.

```
double f(int n) {
    return (n<2)?n:f(n-1)+f(n-2);
}

double i(int n) {
    double x=0, y=1, z;
    for (;n>0;n--){z=x+y; x=y; y=z;}
    return x;
}

typedef struct {double a,b;} Pair;

Pair h(int n) {
    if (n==0) {
        return (Pair) {0,1};
    } else if (n&1) {
        Pair q=h(n-1); double a=q.a, b=q.b;
        return (Pair) {b, a+b};
    } else {
        Pair q=h(n/2); double a=q.a, b=q.b;
        return (Pair) {2*a*b-a*a, a*a+b*b};
   }
}
```

1. Show that the time complexity of `f` is $O(\alpha^n)$ where $\alpha = (1+\sqrt{5})/2$.

   Hint: Use the fact (from week 8) that `f` performs exactly $3f(n+1)-3$ arithmetic operations on an input n along with the fact (from week 5) that $f(n)$ can be written as follows (noting that $\alpha > 1$ and $-1 < \beta < 0$):

   $$f(n) = \frac{1}{\sqrt{5}}(\alpha^n - \beta^n) \text{ where } \alpha = \frac{1}{2}(1+\sqrt{5}) \text{ and } \beta = \frac{1}{2}(1-\sqrt{5})$$

2. Show that the worst case runtime complexity of `h` is $O(\,log_2\, n)$.

   Hint: Use the fact (from week 7) that $2 + 10\,log_2\, n$ is an upper bound on the number of arithmetic operations performed by `h` for all $n > 0$ to show that `h` is $O(2 + 10\,log_2\, n)$. Then show that $2 + 10\,log_2\, n$ is $O(\,log_2\, n)$ and use the transitivity of Big-Oh.

3. Sketch the graphs of 1.6^n, n and $log_2\,(n)$ and determine which functions will always be most efficient in the long run. Run some experiments to see what values of n each of them can compute in practice.

4. **(Optional)** Show the following complexity results:

   (i) $n! = O(n^n)$

   (ii) $log(n!) = O(log(n^n))$.

ANSWERS

1.
$$\gamma(n) = 3f(n+1) - 3 = \frac{3}{\sqrt{5}}(\alpha^{n+1} - \beta^{n+1}) - 3$$

$$\delta(n) = \alpha^n$$

$$\lim_{n\to\infty} \frac{\gamma(n)}{\delta(n)} = \lim_{n\to\infty} \left(\frac{3}{\sqrt{5}} \frac{(\alpha^{n+1} - \beta^{n+1})}{\alpha^n} - \frac{3}{\alpha^n} \right)$$

$$= \lim_{n\to\infty} \left(\frac{3\alpha}{\sqrt{5}} - \frac{\beta^{n+1}}{\alpha^n} - \frac{3}{\alpha^n} \right)$$

$$= \lim_{n\to\infty} \frac{3\alpha}{\sqrt{5}} \quad \text{because} \lim_{n\to\infty} \alpha^n = \infty \text{ and } \lim_{n\to\infty} \beta^{n+1} = 0$$

$$= \frac{3}{2\sqrt{5}}(1+\sqrt{5})$$

which is clearly a finite constant (≈ 2.17)

thus `f` is $O(\alpha^n)$ where $\alpha \approx 1.62$

2. Since $\gamma(n) \leq 2 + 10\, log_2\, n$ and $1 < 2 + 10\, log_2\, n$ for all $n > 0$ it follows

$$\lim_{n\to\infty} \frac{\gamma(n)}{2 + 10\, log_2\, n} \leq 1$$

Thus `h` is $O(2 + 10\, log_2\, n)$.

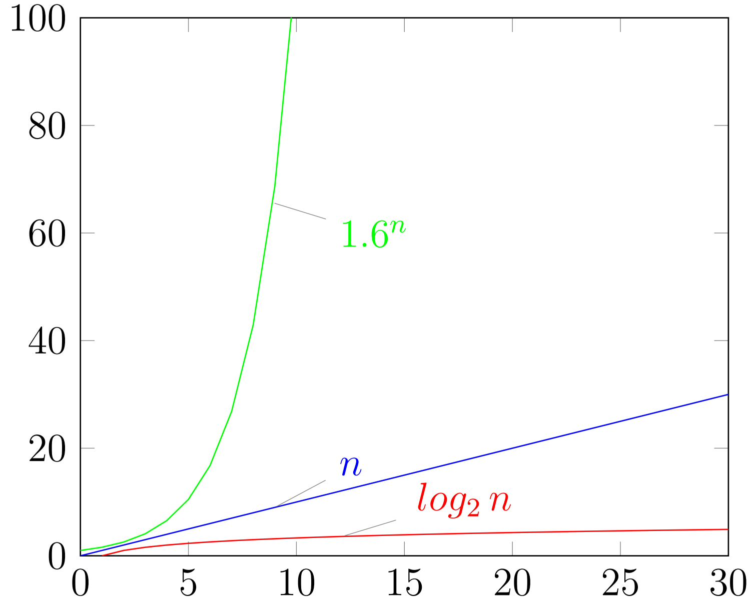
On the other hand

$$\lim_{n\to\infty} \frac{2 + 10\, log_2\, n}{log_2\, n} = \lim_{n\to\infty} \left(\frac{2}{log_2\, n} + 10 \right) = 10 \quad \text{as} \lim_{n\to\infty} log_2\, n = \infty$$

Thus $2 + 10\, log_2\, n$ is $O(\, log_2\, n)$.

Hence `h` is $O(\, log_2\, n)$ by transitivity.

3. Function `h` will always outperform `i` and `f` for large enough n:



4. (i) **Hint:**

$$\frac{n!}{n^n} = \frac{(n)}{n}\frac{(n-1)}{n}\cdots\frac{2}{n}\frac{1}{n}$$

and since all the terms are between 0 and 1 so must the product be.

(ii) **Hint:**

$$log(n!) = log((n)(n-1)\dots(2)(1))$$

$$= log(n) + log(n-1) + \cdots + log(2) + log(1)$$

which is less (term by term and therefore overall) than

$$log(n) + log(n) + ... + log(n) + log(n)$$

$$= nlog(n) = log(n^n)$$