

Frank Moore
Math 918, Taught by Jon-Lark Kim
Homework 1
Week of July 14

Problem 1: Convert $\sqrt{5} = 2.236067$ to a binary decimal.

Solution: $\sqrt{5} = 10.0011110001101\dots$ It is well known how to compute the binary form of the number to the left of the decimal place. The method of writing the decimal to the left of the decimal place may be summed up in the following algorithm below:

```

SET VARIABLE DECIMAL = INPUT.
WHILE ((DECIMAL  $\neq$  0) AND (DECIMAL  $\neq$  1))
    DECIMAL = DECIMAL * 2
    IF (DECIMAL  $\geq$  1) THEN
        PRINT 1
        DECIMAL = DECIMAL - 1
    ELSE
        PRINT 0
    END IF
END WHILE

```

Problem 2: Suppose $\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = L$. Prove that $f \in O(g)$.

So, given $\epsilon > 0$, $\exists N$ such that $n \geq N$ implies that $|\frac{f(n)}{g(n)} - L| < \epsilon$. So, expanding this inequality out, we have $-\epsilon < \frac{f(n)}{g(n)} - L < \epsilon$, which gives $|\frac{f(n)}{g(n)} - L| < \epsilon$. Hence, we have the following: $\frac{f(n)}{g(n)} < L + \epsilon$, which implies that $f(n) < (L + \epsilon)g(n)$. This satisfies our second definition of $O(g)$. Now for the first, set $N_0 = \max(f(1), f(2), \dots, f(N - 1))$. We know by the archimedian property of numbers that there exists an L_0 such that $N_0 < L_0 g(n)$ for all $1 \leq n < N$. Now set $L' = \max(L_0, (1 + L))$. Then by definition of L' , we have that $f(n) < L'g(n)$ for all $n \in \mathbb{N}^+$, which satisfies our first definition of $O(g)$, as desired.

Problem 3: Find the time estimate for multiplying an $r \times n$ matrix with an $n \times s$ matrix where the matrix entries are bounded by m .

Note that each entry involves a sum of the form:

$$a_{i1}b_{1j} + a_{i2}b_{2j} + \dots + a_{in}b_{nj}$$

This takes $O(n \log^2 m + 2(n - 1) \log nm) = O(n(\log^2 m + \log n))$ time. Indeed, the matrix entries are bounded by m , so the n multiplications take $\log^2 m$ bit operations, and the summands are bounded by $n^2 m$, thus the time for an addition takes $\log n^2 m$ bit operations. Note that we have rs of the above sums to compute. Therefore, our total time estimate is that the matrix multiplications requires $O(rsn(\log^2 m + \log n))$ bit operations.

Problem 4: Find a simple algorithm to find $\lfloor \sqrt{n} \rfloor$ in $O(\log^3 n)$ bit operations.

Suppose n is a k -bit integer. Set $m = 2^{\lfloor (k+1)/2 \rfloor}$, which we know is less than the square root of n , while $2^{\lfloor (k+1)/2 \rfloor + 1}$ is greater than the square root of n . Next, for each i , $1 \leq i \leq k$, set $l = m + 2^{\lfloor (k+1)/2 \rfloor - i}$. if

$l^2 > n$, leave m unchanged. Else, set $m = l$, and continue this process for the i mentioned above. This takes $O(\log n)$ iterations and each multiplication test takes $O(\log^2 n)$ bit operations, giving a total time estimate of $O(\log^3 n)$ bit operations.

Problem 5a: Show that the number of bit operations it takes to perform a division $a = qb + r$ is $O((\log b)(1 + \log q))$.

When performing long division while dividing b by a , we see that each subtraction requires $O(\log b + 1)$ bit operations, and that we have at most $O(\log q + 1)$ subtractions that occur, we see that the number of bit operations to obtain the quotient and remainder is $O((\log b + 1)(\log q + 1)) = O((\log b)(\log q + 1))$.

Problem 5b: Use a) to show that the total number of bit operations required for the Euclidean Algorithm is $O((\log b)(\log a))$.

So, each stage of the Euclidean Algorithm requires a division of the form: $r_{i-2} = q_i r_{i-1} + r_i$, for $i = 1, \dots, N$ where $N = 2\lfloor \log a \rfloor + 1$, and we set $r_{-1} = a$ and $r_0 = b$. Therefore, the total time is:

$$\begin{aligned}
& O\left(\sum_{i=0}^N (\log r_{i-1})(\log q_i + 1)\right) \\
&= O\left(\sum_{i=0}^N (\log r_{i-1})(\log q_i + 1)\right) \\
&= O\left(\sum_{i=0}^N (\log r_{i-1} \log q_i \log r_{i-1})\right) \\
&= O\left(\sum_{i=0}^N \log r_{i-1} + \sum_{i=0}^N \log r_{i-1} \log q_i\right) \\
&= O\left(\sum_{i=0}^N \log b + \log b \log\left(\prod_{i=0}^N q_i\right)\right)
\end{aligned}$$

Now, note that working backwards from the Euclidean algorithm we can obtain the fact that $\prod_{i=0}^N q_i < a$.

Therefore, we get:

$$\begin{aligned}
&= O\left(\sum_{i=0}^N \log b + \log b \log a\right) \\
&= O(N \log b + \log b \log a) \\
&= O((2\lfloor \log a \rfloor + 1) \log b + \log b \log a) \\
&= O(\log b \log a)
\end{aligned}$$

Problem 6: Find the smallest nonnegative solution of the following system of congruences: $x \equiv 2(\text{mod}3)$, $x \equiv 3(\text{mod}5)$, $x \equiv 4(\text{mod}11)$, $x \equiv 5(\text{mod}16)$.

Following the proof of the Chinese Remainder Theorem, we have that the lowest nonnegative solution of $x = \sum_i a_i M_i N_i$ modulo $m_1 \cdots m_n$, where $M_i = m_1 \cdots \hat{m}_i \cdots m_n$ (where \hat{m}_i denotes remove m_i from the product), and N_i is the inverse of M_i modulo m_i . Thus, computing $M_1 = 880, M_2 = 528, M_3 = 240, M_4 = 1645, N_1 = 1, N_2 = 2, N_3 = 5, N_4 = 13$, we see that $x = 20453$. Reducing this modulo $m_1 \cdots m_n = 2640$, we see that $x = 1973$ is the smallest nonnegative solution to the system above.

Problem 7: Let $m = 7785561297230017200 = 2^4 \cdot 3^4 \cdot 5^2 \cdot 7 \cdot 11 \cdot 13 \cdot 19 \cdot 31 \cdot 37 \cdot 41 \cdot 61 \cdot 73 \cdot 181$. Find $6647^{362} \pmod{m}$.

Let $g = \text{lcm}(\{\phi(p_i^{\alpha_i})\})$. Note that if $a' \equiv a \pmod{g}$ and $(b, m) = 1$, then a proposition in class says that $b^g \equiv b^{a'} \pmod{m}$. Note that the above lcm is 360. Thus, $6647^{362} \equiv 6647^2 \pmod{m}$, so that $6647^{362} \equiv 44182609 \pmod{m}$.

Problem 8a: Prove that $\prod_p \frac{1}{1-1/p}$ diverges to infinity where the product is over all primes p .

Note that $\sum_n = 0^\infty \frac{1}{p^n} = \frac{1}{1-1/p}$. Therefore, we have that:

$$\prod_p \frac{1}{1-1/p} = \prod_p \sum_n \frac{1}{p^n} = 0^\infty \frac{1}{p^n}$$

Now, when expanding this product, note that we get every integer in the denominator once and only once, by the Fundamental Theorem of Arithmetic. Therefore, the above product becomes the well known harmonic series, $\sum_n \frac{1}{n}$, which is well known to diverge.

Problem 8b: Using a), prove that $\sum_p \frac{1}{p}$ diverges.

Claim: For $0 \leq x \leq 1/2$, $x > -1/2 \log(1-x)$. To see this, one need only observe the graph of these functions on a graphing utility. Therefore, we have that:

$$\sum_p 1/p > -1/2 \sum_p \log(1-1/p) = -1/2 \sum_p \log(p-1) - \log p$$

On the other hand, note that $\log(\prod_p \frac{1}{1-1/p}) = \sum_p \log(\frac{1}{1-1/p}) = \sum_p \log(p/(p-1)) = -\sum_p \log(p-1) - \log p$.

Combining these two facts, we have that $2 \sum_p 1/p$ is bounded below by the logarithm of a divergent product, hence the sum of reciprocals of primes diverges.

Problem 8c: Prove that there exists a sequence of integers $\{n_j\}$ such that $\lim_{n \rightarrow \infty} \frac{\phi(n_j)}{n_j} = 1$ and $\lim_{n \rightarrow \infty} \frac{\phi(n_j)}{n_j} = 0$.

Take n_j to be the sequence of primes, then $\lim_{n \rightarrow \infty} \frac{\phi(n_j)}{n_j} = \lim_{n \rightarrow \infty} \frac{n_j-1}{n_j} = 1$. For the second limit, simply take $n_j = j!$. Since the factorial function grows much quicker than the primes to by the prime number theorem, we have that $\lim_{n \rightarrow \infty} \frac{\phi(n_j)}{n_j} = 0$.

Problem 9: Prove that if $2^n - 1$ is prime then n must be prime. Also, prove that if $2^n + 1$ is prime then n is a power of 2.

Suppose $n = ab$, with $a, b \geq 2$. Then by a proposition, we have that $2^n - 1 = (2^a - 1)t$ for some t . Since a is at least 2, we have that $2^a - 1$ is not prime, a contradiction. For the second proposition, suppose n has an odd factor, say a . Then we have $(2^n + 1)/(2^a + 1) = 2^{(k-1)a} - 2^{(k-2)a} + \dots + 1$.