Keith and the Big Bad Sequence

May 26, 2016

We need to check whether $\binom{m}{n}$ is even. A direct calculation of $\binom{m}{n} = \frac{m(m-1)(m-2)...(m-n+1)}{n!}$ is probably too slow - it is linear time in n and the numbers get prohibitively large.

Lucas's Theorem allows us to compute $\binom{m}{n} \pmod 2$ quickly, working in $O(1)$ time. If we let $m_k m_{k-1} m_{k-2} ... m_0$ and $n_k n_{k-1} n_{k-2} ... n_0$ be the binary representations of m and n, respectively, where the smaller number is padded with zeroes on the left side so that each representation has the same number of digits. Lucas's Theorem tells us that $\binom{m}{n} \equiv \binom{m_k}{n_k}\binom{m_{k-1}}{n_{k-1}}...\binom{m_0}{k_0} \pmod 2$.

The only way for this product to yield 0 modulo 2 (so that $\binom{m}{n}$ is even) is if at least one of the m_i digits is less than the corresponding n_i digit - that is, there is at least one index i such that $m_i = 0$ and $n_i = 1$. We can check this condition using bitwise operators. Consider the expression $\sim m_i \ \& \ n_i$. This expression yields 1 if and only if $m_i = 0$ and $n_i = 1$. Otherwise, it evaluates to 0. Therefore, if there are no indices i such that $m_i = 0$ and $n_i = 1$, every such expression $\sim m_i \ \& \ n_i$ evaluates to 0, so that the general $\sim m \ \& \ n = 0$, and the combination is odd. On the other hand, if there is at least one such index (meaning the combination is even), then one of those expressions will evaluate to 1, so that $\sim m \ \& \ n! = 0$. Thus, we have shown that the boolean $\binom{m}{n} \ \% \ 2 == 0$ is equivalent to the boolean $\sim m \ \& \ n \ != 0$, which works in constant time. So we have found a convenient condition to check for whether $\binom{m}{n}$ is even.

All that remains is to iterate over the sequence over all possible lengths and keep track of the maximum length and the index associated with it. Sample code in java is provided below:

```
int maxLength = 0;
int maxIndex = 0;
```

```
outerloop:
for (int length = 0; length < N; length++) {
        for (int start = 0; start < N - length; start++) {
                for (int index = start; index < start + length; index++
                        if (~seq[index] & length == 0) {
                                start = index;
                                break;
                        }
                        if (index == start + length - 1) {
                                maxLength = length;
                                maxIndex = start;
                                continue outerloop;
                        }
                }
        }
}
System.out.println(maxLength + " " + maxIndex);
```

Remark. *There are other ways to solve this problem than Lucas's Theorem. Another way to determine whether* $\binom{m}{n}$ *is even is to express it in factorial form,* $\frac{m!}{n!(m-n)!}$ *and compare the number of 2's in the numerator and in the denominator. If the number of 2's (that is, the greatest power of 2 that divides the numerator) in the numerator is greater, the resulting binomial is even. To compute the greatest power of 2 that divides* n*, the well known Legendre's Formula gives us the formula* $\sum_{i=1}^{\infty} \lfloor \frac{n}{2^i} \rfloor$*. It is somewhat computationally expensive but can work. Better is the other form of Legendre's:* $\frac{n-S_p(n)}{p-1}$*, where* $S_p(n)$ *is the sum of the digits of* n *in base-*p*, and* $p = 2$ *in this case. This is actually a calculation with constant time complexity so it operates with the same time complexity as the original solution.*