Introduction to Algorithms and Informatics April 14, 2010

Lecture 1

Lecturer: David Avis *Scribe: Kazuhisa Seto*

In this class, we introduce the basics of "algorithms" and "informatics". In this lecture, we study the history and the definition of algorithm and one of the basic algorithm: Euclid's algorithm.

1 What is an algorithm?

The word "algorithm" has an interesting origin that [1] describes as follows:

"The word algorithm comes from the name of the 9th century Persian Muslim mathematician Abu Abdullah Muhammad ibn Musa Al-Khwarizmi. The word algorism originally referred only to the rules of performing arithmetic using Hindu-Arabic numerals but evolved via European Latin translation of Al-Khwarizmi's name into algorithm by the 18th century. The use of the word evolved to include all definite procedures for solving problems or performing tasks."

There exist many definition of algorithm. An intuitive and simple definition of algorithm was introduced by Knuth. Knuth says an algorithm should have the following properties:

1: Finite $\cdots$ not necessary for probabilistic algorithm
2: Definite $\cdots$ each step clearly specified
3: Input
4: Output
5: Effectiveness $\cdots$ could be computed (in principle) by a human with pen and paper
6: Correctness $\cdots$ output must be correct.

Input and Output are very important when we try to design the algorithms.

2 Euclidean algorithm

The Euclidean algorithm is considered once of the first algorithms, and is still one of the best.

Algorithm 1 Euclidean algorithm : GCD(a, b)

Input: Two integer a, b $(a > b)$
Output: Largest integer k that divides both a and b

1: Divide the number a by b, the remainder is r
2: Replace a by b
3: Replace b by r
4: Continue until a can't be more divided. In this case, a is the gcd.

It is a very efficient algorithm to find the largest divisor for two integers.

Example 1 $\mathrm{GCD}(96, 20) = 4$

1: $\mathrm{GCD}(96(= a), 20(= b))$

$$\frac{96}{20} = 4 + \frac{16(= r)}{20}$$

2: $\mathrm{GCD}(20(= a), 16(= b))$

$$\frac{20}{16} = 1 + \frac{4(= r)}{16}$$

3: $\mathrm{GCD}(16(= a), 4(= b))$

$$\frac{16}{4} = 4 + \frac{0(= r)}{16}$$

Outputs 4

Claim 1 *gcd(a,b) = gcd(b,r)*

Proof Suppose $a > b$ and that k is the largest divisor of a and b. Let $a = kt$ and $b = ks$ for integers s and t. Since $a > b$ we can divide b into a getting an integer q and a remainder r. Therefore, we can write $a = bq + r$ for an integer $1 <= q < b$ and $0 <= r < b$. So $kt = ksq + r$, and $t = sq + \frac{r}{k}$. $\frac{r}{k}$ is an integer because both t and sq are integers. Thus, $r = ku$ for some integer u. This concludes the proof. ■

So each step of the algorithm has the same gcd as the original problem, gcd(a,b). Is this a good algorithm? "Good" means this algorithm outputs the answer "efficiently". We consider another algorithm for gcd such as the following:

Algorithm 2 Bad GCD algorithm : $\mathrm{GCD}_{bad}(a, b)$

Input: Two integer a, b $(a > b)$
Output: Largest integer k that divide with a and b

1: Calculate $a - b(= s)$
2: Replace a by b or s (whichever is greater)
3: Replace b by b or s (whichever is smaller)
4: Continue until $b = 0$. In this case, a is the gcd.

Let see an example of this gcd algorithm.

Example 2 $\mathrm{GCD}_{bad}(96, 20)$

$$\begin{aligned} GCD_{bad}(96, 20) &= GCD_{bad}(76, 20) = GCD_{bad}(56, 20) = GCD_{bad}(36, 20) = GCD_{bad}(20, 16) \\ &= GCD_{bad}(16, 4) = GCD_{bad}(12, 4) = GCD_{bad}(8, 4) = GCD_{bad}(4, 0) \\ &= 4 \end{aligned}$$

It is easy to see that GCD algorithm is more efficient than GCD_{bad}. GCD takes 3 steps to find the greatest divisor for 96 and 20 in Example 1, on the contrary, GCD_{bad} takes 8 steps to do so in Example 2. In fact, Euclidean algorithm is known to be very efficient algorithm.

Now, we consider how efficient Euclidean algorithm is, that is, "How many steps does Euclidean algorithm take? "What are two smallest values of a and b so that Euclidean algorithm takes n steps? $(a > b > 0)$."

They exist for any n. They are Fibonacci numbers, F_{n+1}, F_{n+2}.

2.1 Fibonacci Numbers

The Fibonacci numbers are as follows:

1 1 2 3 5 8 13 21 34 $\cdots$

This string of the numbers is formulated into the following equation.

$$F_{n+2} = F_{n+1} + F_n,\ F_1 = 1,\ F_2 = 1$$

We consider Euclidean algorithm for F_{n+2} and F_{n+1} and show the following claim.

Claim 2 *For each $n \geq 1$, $GCD(F_{n+2}, F_{n+1}) = 1$, and it takes the Euclidean algorithm n steps to compute this.*

Proof If $n = 1$, then GCD(F_3, F_2) = GCD(2,1) =1 and the Euclidean algorithm takes $n = 1$ step. For $n \geq 2$, $\frac{F_{n+2}}{F_{n+1}} = 1 + \frac{F_n (=r)}{F_{n+1}}$. So, GCD$(F_{n+2}, F_{n+1})$ = GCD(F_{n+1}, F_n). By repeating this $n-1$ times, GCD(F_{n+2}, F_{n+1}) = GCD(F_3, F_2). Hence with one additional step we conclude GCD$(F_{n+2}, F_{n+1}) = 1$. Thus our claim holds. ■

Now, we show the following claim.

Claim 3 *For $n \geq 1$, F_{n+2} and F_{n+1} are two smallest values of a and b so that the Euclidean algorithm takes n steps $(a > b > 0)$.*

Proof (by induction)
Base Case Let $n = 1$, then $a = 2(= F_3)$ and $b = 1(= F_2)$ and the claim holds. Let $n = 2$, then $b \geq 2$, so $a \geq 3$. GCD$(3, 2)$ = GCD$(2, 1)$ = GCD$(1, 0) = 1$. Euclidean algorithm takes $2(= n)$ steps, thus claim holds by $(a =)3 = F_4, (b =)2 = F_3$.
Induction Step Assume that smallest $a > b$ for n steps are F_{n+2} and F_{n+1}, respectively. For $n+1$ steps, let $a > b$ be the smallest integers requiring $n + 1$ steps. Consider the first step of Euclid's algorithm. We have GCD(a, b) = GCD(b, r) where $a = qb + r$ and we know that GCD(b, r) takes n steps. Since a and b are smallest, we must have $q = 1$, so $a = b + r$. Again, since a is smallest, b and r must be the smallest integers that require n steps in Euclid's algorithm. By induction, $b = F_{n+2}$ and $r = F_{n+1}$, thus $a = F_{n+2} + F_{n+1} = F_{n+3}$. This concludes the proof. ■

Next, we consider the following question:

Problem 4 *If $a > b$ and a has n digits, what is the maximum number of steps required by Euclidean algorithm?*

To solve this problem, we first ask "How many digits does F_n have ?"

We show the following claim.

Claim 5 *F_{6n+2} has at least $n + 1$ digits, for each $n \geq 0$.*

Proof We compute Fibonacci number recursively as follows.

$$\begin{aligned}
F_{n+1} &= F_{n+1} \\
F_{n+2} &= F_{n+1} + F_n \\
F_{n+3} &= F_{n+2} + F_{n+1} = 2F_{n+1} + F_n \\
F_{n+4} &= F_{n+3} + F_{n+2} = 2F_{n+1} + F_n + F_{n+1} + F_n = 3F_{n+1} + 2F_n \\
F_{n+5} &= 5F_{n+1} + 3F_n \\
F_{n+6} &= 8F_{n+1} + 5F_n \geq 10F_n
\end{aligned}$$

Thus, F_{n+6} has at least one more digit than F_n. Since F_2 has one digit, F_8 has at least two digits, F_{14} has at least three digits, etc., and the claim holds. ■

We can now answer Problem 4 by combining the Claims 3 and 5.

Claim 6 *Euclid's algorithm takes strictly less than* $6n$ *steps if* a *has* n *digits.*

Proof By contradiction. Suppose a has n digits and takes at least $6n$ steps using the Euclidean algorithm. By Claim 3 a must be at least as big as F_{6n+2}. But by Claim 5, F_{6n+2} has at least $n+1$ digits, so a must also. This is a contradiction. ■

3 Conclusion

In this lecture, we studied the history of algorithms and Euclid's algorithm to find the greatest common divisor for two integers. We showed the Euclidean algorithm is extremely efficient, since it requires at most a linear number of steps in terms of the number of digits of the input numbers.

References

[1] Scriptol. "http://www.scriptol.com/programming/algorithm-history.php"