Bob Burton

December 16, 2003

Newer Math Honors Seminar

GCD via the Euclidean Algorithm

History and Background Information

The Euclidean algorithm, also known as Euclid's algorithm, is one of the oldest known algorithms, dating back to around 300 B.C. It is believed that Euclid developed the algorithm in a geometric context because he wanted to solve the problem of finding a common "measure" for the lengths of two lines [1]. Today, one would say that he was trying to find the greatest common divisor, or GCD, for the lengths of the lines. His algorithm, which did not involve factoring, proceeded by repeated subtraction of the shorter segment from the longer one [1] (instead of constant division and subtraction). The algorithm appeared in Proposition 2 of Book VII (the first of four on numbers and arithmetic) in the set of *The Thirteen Books of the Elements* [2].

Purpose and Efficiency

The purpose of the Euclidean algorithm is to compute the greatest common divisor of two natural numbers [3]. These numbers will be called a and b, where $a > b$ for simplicity. The efficiency is concerned with the longest possible running time of the algorithm [4]. The "worst-case" scenario for the algorithm occurs with two consecutive Fibonacci numbers, which requires $\Theta(n)$ divisions, where n is the number of digits of the input [5]. This is approximately equal to the number of sequential additions it takes to obtain the larger Fibonacci number of which the GCD is being computed (for example, if *a* is the 125th Fibonacci number, it can take up to 125 steps to obtain the GCD of *a* and *b*).

How the Algorithm Works

There are several ways to look at how the algorithm works. The first way is as follows:

- if $b = 0$, then return *a*

- otherwise, return the GCD of *b* and *a* mod *b* [6]. Repeat until left with the task of finding the GCD of a natural number and 0.

Another way to view the algorithm is as follows:

1. If $a|b$ (*a* divides *b*), then the GCD of *a* and *b* [gcd(*a*,*b*) for short] is *a*. This is true because no number may have a divisor greater than itself.
2. If $a = bu + r$, where *u* and *r* are integers, then $\gcd(a,b) = \gcd(b,r)$ [6].

It is not completely obvious why this is true, so this must be explained. Every common divisor of *a* and *b* divides *r*. However, $\gcd(a,b)|b$. Therefore, gcd(*a*,*b*) is a common divisor of both *b* and *r*, so $\gcd(a,b) \leq \gcd(b,r)$. The reverse is also true because every divisor of *b* also divides *a* [6].

It is important to recognize that the algorithm must eventually terminate because every step of the algorithm reduces the integers to smaller ones. Since these integers are finite, eventually the process must end [6].

Consider the following simple example:

a = 2958 b = 198

2958 = 198*14 + 186	r = 186	so, gcd(2958,198) = gcd(198,186)
198 = 186*1 + 12	r = 12	so, gcd(198,186) = gcd(186,12)
186 = 12*15 + 6	r = 6	so, gcd(186,12) = gcd(12,6)
12 = 6*2 + 0	r = 0	so, gcd(12,6) = 6

Therefore, gcd(2958, 198) = 6.

Now, let's look at a more difficult example involving longer numbers. Though it is a little more complex, it could still easily be computed by hand:

a = 6385720 b = 471328

6384720 = 471328*13 + 258456	so, gcd(6385720,471328) = gcd(471328,258456)
471328 = 258456*1 + 212872	so, gcd(471328,258456) = gcd (258456,212872)
258456 = 212872*1 + 45584	so, gcd(258456,212872) = gcd(212872, 45584)
212872 = 45584*4 +30536	so, gcd(212872,45584) = gcd(45584,30536)
45584 = 30536*1 + 15048	so, gcd(45584,30536) = gcd(30536,15048)
30536 = 15048*2 + 440	so, gcd(30536,15048) = gcd(15048,440)
15048 = 440*34+88	so, gcd(15048,440) = gcd(440, 88)
440 = 88*5 + 0	so, gcd(440,88) = 88

Therefore, gcd(6385720,471328) = 88. It is evident from this example that even when numbers are large, the algorithm is still very efficient. Here, only eight steps were needed to calculate the GCD of two numbers of six and seven digits, which is not significantly longer than the first example.

Corollary and Proof

A corollary has been established that for every pair of whole numbers a and b, there exist two integers s and t such that $as + bt = \gcd(a,b)$. The proof of this is done by induction. We have already assumed that $a > b$. First, let's establish a representation. Let Eulen(a,b) denote the length (number of steps) of the algorithm for the pair a,b [So, Eulen(2958,198) = 4]. If Eulen (a,b) = 1, then $a = b*u$ for an integer u. So, $a + (1 - u)*b = b = \gcd(a,b)$. Then, $s = 1$ and $t = 1 - u$ [because $\gcd(a,b) = as + bt$]. Let Eulen(a,b) = n and assume that the corollary is established for all pairs of numbers for Eulen < n. Apply one step of the algorithm: $a = bu + r$, Eulen(b,r) = n-1. By the inductive assumption, there are an x and y that exist such that

$bx + ry = \gcd(b,r) = \gcd(a,b)$. We can express r as $r = a - bu$ from our original equation that $a = bu + r$. If we multiply through by y, we obtain that $ry = ay - buy$. Substituting for ry in our other equation and simplifying, we obtain:

$bx + ay - buy = \gcd(a,b)$

$b(x - uy) + ay = \gcd(a,b)$

Now, take $s = x - uy$ and $t = y$ [6].

Notes and Remarks

It should be noted that any linear combination $as + bt$ is divisible by any common factor of a and b; particularly, any common factor also divides $\gcd(a,b)$. The reverse is also true. Any linear combination $as + bt$ is divisible by $\gcd(a,b)$. GCD(a,b) is the least positive integer representable in the form $as + bt$. All others are multiples of the GCD of a and b [6].

The generalization of the corollary into an arbitrary field is known as either Bézout's Identity or Bézout's Lemma, named after the French mathematician Éttiene Bézout, who lived from 1730 to 1783. A *field* is a ring in which multiplication is a group operation. A *ring* is an additive commutative group in which a second operation is also defined [6].

Coprime numbers

Two natural numbers a and b are said to be coprime, or relatively prime, if they share no common positive factors other than 1 [6]. Therefore, this occurs when $\gcd(a,b) = 1$. This may also be expressed as $a \perp b$.

The Fundamental Theorem of Arithmetic

The Fundamental Theorem of Arithmetic states that any integer N may be expressed as some product of its prime factors: $N = p_1^{n1} * p_2^{n2} * \ldots * p_m^{nm}$, where the p_i's are prime numbers and n_i's are positive [7]. This is used to help define some properties of the GCD.

The Extension of Euclid's Algorithm

There is an extension of Euclid's algorithm which is designed to compute the values of *s* and *t* mentioned earlier. Let's begin stating some obvious facts. Clearly, $a = 1*a + 0*b$ and $b = 0*a + 1*b$. To apply the extension of the algorithm, write these equations in a table, with columns for the left side of the equation, the number multiplied by *a*, and the number multiplied by *b*. Next, apply Euclid's algorithm to the left side of the equation. Assume $a = bu + r$ and multiply the second equation by *u* and subtract this equation from the first:

$a = 1*a + 0*b$ remains $\qquad a = 1*a + 0*b$

$b = 0*a + 1*b$ becomes $\qquad \underline{-bu = 0*a - u*b}$

$$r = a - bu = a - ub$$

Therefore, $r = 1*a - ub$ [8].

Next, apply the same procedure to the last two equations. Continue in this manner until the Euclidean algorithm can no longer be applied to the left side of the equation. Use the conventional method of solving linear equations by omitting all terms in a linear combination except for the left side and the two coefficients on the right. Place the results in the table, except with a fourth column representing *u* (from $a = bu + r$, which changes every step). Multiply the three other numbers in the row to the left of *u* by *u* and subtract them from the numbers in the row directly above. Record the results on the next line and repeat the process [8]. The procedure terminates when a 0 is reached on the left side of the equation.

Consider our first example:

a = 2958 *b* = 198

Left-side	*a	*b	u
2958	1	0	-
198	0	1	14
186	1	-14	1
12	-1	15	15
6	16	-239	2
0	STOP!		

This shows us that 16*2958 -239*198 = 6. Note that 16 and 239 are coprime. This must be true for any set of *a* and *b*; otherwise, if *s* and *t* were not coprime, the entire equation (including the right side) could be divided by the common factor shared by *s* and *t*, resulting in a new, smaller GCD. In this example, division by gcd(*a*,*b*) results in 16*493 – 239*33 = 1. From this, we obtain that 239/493 and 16/33 are consecutive fractions in the Farey series of order 493.

Notes and Remarks

In the extension of the Euclidean algorithm, there are only two additional multiplications and two additional subtractions in each step. Because these operations do not affect the progress of the original algorithm, it proves that the extended algorithm terminates with the original [8]. However, the extended algorithm is far more reliable than the original. Assume the algorithm furnished *s*, *t*, and *g*, such that $as + bt = g$. We can verify whether $g|a$ or $g|b$. This implies that $g|\gcd(a,b)$ and because *g* is expressed as a linear combination of *a* and *b*, it must be true that $\gcd(a,b)|g$. Therefore, $g = \gcd(a,b)$. By finding *s* and *t*, the algorithm simply proves that it is correct.

The Farey Series

The Farey Series, F_N, is the set of all fractions, in lowest terms, between 0 and 1 whose denominators do not surpass N, arranged in order of increasing magnitude [9]. For example, F_6 is 0/1, 1/6, 1/5, ¼, 1/3, 2/5, ½, 3/5, 2/3, ¾, 4/5, 5/6, 1/1. The "N" is considered the order of the series. When using the Extended algorithm, two consecutive Farey series fractions are obtained by dividing the equation by the GCD and then comparing $s/(b/\gcd(a,b))$ and $t/(a/\gcd(a,b))$.

Binary Euclid's Algorithm

The binary algorithm was discovered in 1962 by R. Silver and J. Tersion. It was published five years later by G. Stein in 1967. It can be proved inductively, but that proof will not be shown. It follows the same basic concept of the original algorithm, except it only uses 0's and 1's because it is specifically designed for computer use. It is a slight improvement to the original algorithm and it is the first such improvement to it in over 2000 years [10]. All division in the algorithm is by 2, which can be easily implemented on a binary computer.

The formula for Binary Euclid's algorithm is $\gcd(a,b) = \gcd(b, a \bmod b)$, where $a \bmod b$ is the remainder of the division of a by b. The algorithm is based on the postulate that $\gcd(a,0) = a$, along with some other properties of GCD. In my first example, $\gcd(2958,198) = \gcd(198,186) = \gcd(186,12) = \gcd(12,6) = \gcd(6,0) = 6$. Some important properties of GCD are as follows:

1. $\gcd(ca,cb) = c \gcd(a,b)$
2. if $\gcd(a,b) = 1$, then $\gcd(a,bc) = \gcd(a,c)$
3. $\gcd(a,b) = \gcd(a-b,b)$

Properties (1) and (2) are based on the Fundamental Theorem of Algebra. Property (3) is based on the basic properties of modular arithmetic and division [10].

The Binary algorithm for finding gcd(a,b) is based on the following procedure:

1. If a and b are both even, perform a right shift to both a and b because gcd(a,b) = 2 gcd(a/2,b/2), and record a saved factor of 2.

2. If a is even and b is odd, perform a right shift to a because gcd(a,b) = gcd(a/2,b).

   If b is even and a is odd, perform a right shift to b because gcd(a,b) = gcd(a,b/2).

3. If a and b are both odd, replace a with $a - b$ if a is larger, or $b - a$ if b is larger because $|a - b| < \max(a,b)$ since $|a - b|$ is even.

There is a machine instruction known as a *right shift* in which the right most bit is discharged, the remaining bits are shifted by one place to the right, and the leftmost bit is set to 0 [10]. This is equivalent to dividing by 2.

Let's look at our original example:

$a = 2958$ $b = 198$

a	*b*	What to do	Saved factor
2958	198	right shift *a* and *b*	2
1479	99	$a - b$	
1380	99	right shift *a*	
690	99	right shift *a*	
345	99	$a - b$	
246	99	right shift *a*	
147	99	$a - b$	
48	99	right shift *a*	
24	99	right shift *a*	
12	99	right shift *a*	
6	99	right shift *a*	
3	99	$b - a$	
3	96	right shift *b*	
3	48	right shift *b*	
3	24	right shift *b*	
3	12	right shift *b*	
3	6	right shift *b*	

3	3	$a - b$
3	0	STOP!

Since the last nonzero factor obtained is 3, we multiply this by the saved factor of 2 that we have, and obtain that the GCD is 3*2 = 6. Therefore, gcd(2958,198) = 6.

A Corollary Involving Modular Multiplicative Inverses

If p is prime and w is an integer less than p, then gcd(p,w) = 1 and $qp + vw = 1$ (where q and v are integers), so v is the multiplicative inverse of w mod p.

Consider the following basic example:

$p = 79$ $w = 36$ Let's use the extended algorithm.

Left-side	*p	*w	u
79	1	0	-
36	0	1	2
7	1	-2	5
1	-5	11	7
0	STOP!		

Thus, from the algorithm we obtain that 11*36 – 5*79 = 1. Based on the corollary, 11 is the multiplicative inverse of 36 mod 79. This is because of the fact that $qw = 1$ mod p [11].

Euclid's Game

In Euclid's game, there is a board with two numbers on it at the beginning of the game. The two players input the difference of any two numbers on the board. The player unable to make a move at the end of the game is the loser. The two original numbers will be represented by a and b, where $a > b$. The game is based on the idea that the difference of any two numbers is divisible by their GCD. Thus, only numbers obtained by taking the difference are multiples of gcd(a,b). All such numbers must appear, regardless of the sequence that they are inputted into the game. To determine whether it would be advantageous to go first or second, use the formula to determine the total number of numbers on the board: $N = a/\gcd(a, b)$, where N is the total number of numbers on the board at the game's end [12].

The proof of this is by contradiction. To understand why all differences must appear, assume that the game is over and that h is the smallest number present on the board. Then, the collection of numbers on the board coincides with the set A of all multiples of h not exceeding the largest of a and b. We know that $h|a$ and $h|b$. For example, if $a = mh + z$, then we could form differences $a – h$, $a – 2h$, etc. and eventually get on the board $z < h$, which contradicts the minimality of h. Therefore, for some d and k, $a = dh$ and $b = kh$. So, on the board is A $\{ih: i = 1, \ldots, \max(d,k)\}$. Recall that $h|\gcd(a,b)$. But, the difference of any two numbers is divisible by their GCD. Therefore, any number present on the board must be divisible by $\gcd(a,b)$. Particularly, $\gcd(a,b)|h$, so $h = \gcd(a,b)$ [12].

Another explanation directly uses Euclid's algorithm. First, assume $a > b$. Form $a = bu + r$ (r must eventually be obtained because it can be computed by continually subtracting b from a until no longer possible). Next, continue with $b = rf + l$. With a and r on the board, continuously subtract to obtain l. Continue in this manner until the algorithm stops and you will have computed $\gcd(a,b)$ [12].

Demonstrating Euclid's Algorithm Using Rectangles

Euclid's algorithm may be demonstrated by using rectangles [13]. Take the equation $83x + 19y = 1$. That can be broken up into a rectangle of 83 x 19. Then, by breaking the rectangle in 19 x 19 squares, we are left with a rectangle of 19 x 7. That 19 x 19 rectangle may be broken up into squares of 7 x 7 until a rectangle of size 7 x 5 remains. When the 7 x 5 rectangle is broken up into a square of 5 x 5, only a 5 x 2 rectangle remains. This breakdown process continues and the rectangle is broken into 2 x 2 squares and a 2 x 1 rectangle, which is finally broken into squares of 1 x 1. This is a good demonstration of Euclid's algorithm because it follows the same concept.

From this, we can use the last nonzero remainder to find the solution: 1 = 5 – 2*2. Substitute remainders back until we get the first equation with P = 83 and Q = 19. The remainders are in square brackets.

1 = [5] – 2[2]
1 = [5] – 2([7] – [5])
1 = -2[7] + 3[5]
1 = (3*19) – 8[7]
1 = (3*19) – 8(83 – (4*19))
1 = (-8*83) + (35*19)

From this, we obtain that the solution of $83x + 19y = 1$ is that $x = -8$ and $y = 35$. This can be a helpful way to understand Euclid's algorithm when learning its concept for the first time.

Summary

In summary, Euclid's algorithm is the most efficient method of finding the GCD of two integers. Its running time tends to be relatively short and there are several ways to use the algorithm to obtain the correct answer. It can be used to compute the multiplicative inverse of numbers in modular multiplication, also. Though the algorithm is over 2000 years old, it is still very efficient and useful in the field of mathematics.

References

[1] "Euclidean Algorithm." Wikipedia, the Free Encyclopedia. 4 December 2003. <http://en2.wikipedia.org/wiki/Euclidean_algorithm>

[2] Barile, Margherita. "Euclidean." 4 December 2003. <http://mathworld.wolfram.com/Euclidean.html>

[3] Littman, Michael L. "Euclid's Algorithm." 3 December 2003. <http://www.cs.duke.edu/~mlittman/courses/Archive/cps130-97/lectures/lect03/node4.html>

[4] Littman, Michael L. "Euclid's Algorithm." 3 December 2003. <http://www.cs.duke.edu/~mlittman/courses/Archive/cps130-97/lectures/lect03/node5.html>

[5] Littman, Michael L. "Euclid's Algorithm." 3 December 2003. <http://www.cs.duke.edu/~mlittman/courses/Archive/cps130-97/lectures/lect03/node12.html>

[6] Bogomolny, Alexander. "Euclid's Algorithm." 3 December 2003. <http://www.cut-the-knot.org/blue/Euclid.shtml>

[7] Bogomolny, Alexander. "GCD and the Fundamental Theorem of Arithmetic." 3 December 2003. <http://www.cut-the-knot.org/blue/gcd_fta.shtml>

[8] Bogomolny, Alexander. "Extension of Euclid's Algorithm." 3 December 2003. <http://www.cut-the-knot.org/blue/extension.shtml>

[9] Bogomolny, Alexander. "Farey Series." 3 December 2003. <http://www.cut-the-knot.org/blue/farey.shtml>

[10] Bogomolny, Alexander. "Binary Euclid's Algorithm." 3 December 2003.

<http://www.cut-the-knot.org/blue/binary.shtml>

[11] "Euclidean Algorithm, The." 7 December 2003.

<http://math.nmsu.edu/crypto/public_html/EuclideanAlgo.html>

[12] Bogomolny, Alexander. "Euclid's Game." 3 December 2003.

<http://www.cut-the-knot.org/blue/EuclidAlg.shtml>

[13] Beardon, Alan and Toni. "Euclid's Algorithm." 4 December 2003.

<http://www.nrich.maths.org.uk/mathsf/journalf/sept99/art1/index.html>

<http://www.nrich.maths.org.uk/mathsf/journalf/oct99/art1/index.html>