Journal of the Korean Data & Information Science Society
2014, **25**(6), 1467–1474

http://dx.doi.org/10.7465/jkdi.2014.25.6.1467
한국데이터정보과학회지

확장 유클리드 알고리즘에 대한 컴퓨터 집약적 방법에 대한 연구

김대학[1] · 오광식[2]

[1]대구가톨릭대학교 수학과 · [2]대구가톨릭대학교 수학교육과

접수 2014년 10월 9일, 수정 2014년 11월 6일, 게재확정 2014년 11월 11일

요 약

본 논문에서는 정수론 분야에서 가장 기초적인 방법으로 소개되는 유클리드 알고리즘과 이를 확장한 확장 유클리드 알고리즘을 소개하고 이들에 대한 컴퓨터 집약적 방법을 연구하였다. 이들 알고리즘들은 공개키 암호 분야에서 암호화의 과정에서 반드시 거쳐야 하는 과정들 중의 하나로서 그 응용성이 날로 부각되고 있다. 확장 유클리드 알고리즘에 대한 컴퓨터 집약적 방법으로서 마이크로소프트 엑셀과 C 언어를 이용하는 두 가지 방법을 각각 고안하여 제안하였다. 본 논문은 단순히 정수론 차원의 계산을 쉽고 편리하게 하기 위함만이 목적이 아니고 아주 큰 수에 대한 역원 (곱셈에 대한 역원)의 계산과 이의 공개키 암호화 활용에서 의의를 찾을 수 있다.

주요용어: 곱셈에 대한 역원, 모듈로 연산, 최대공약수, 컴퓨터 집약적 방법, 확장 유클리드 알고리즘.

1. 서론

두 정수의 최대공약수 g를 구하는 일은 그들의 공약수를 나열한 후 가장 큰 수를 고르는 방법으로도 구할 수 있다. 그러나 아주 큰 수에 있어서는 꾀나 지루하고도 성가신 일이 될 것이다. 다행스럽게도 2000년전 수학자 유클리드가 나눗셈 정리 (division algorithm)를 활용하는 방법 즉 유클리드 알고리즘 (Euclidean algorithm)을 고안했다. 이 방법이 보다 효과적으로 최대공약수를 구하는 방법이다.

유클리드 알고리즘은 주어진 두 정수 a, b $(a > b)$를 서로 나누어 몫 (quotient) q와 나머지 (remainder) r을 얻고 다시 b와 r을 서로 나누어 또 다시 몫과 나머지를 얻는 방법을 나머지가 0이 될 때까지 계속 반복하여 나머지가 0이 되기 직전의 나머지를 최대공약수로 얻는 방법이다. 이를 정리하면 다음과 같다.

$$\begin{aligned} a &= q_1 b + r_1, 0 < r_1 < b \\ b &= q_2 r_1 + r_2, 0 < r_2 < r_1 \\ r_1 &= q_2 r_2 + r_3, 0 < r_3 < r_1 \\ &\vdots \\ r_{n-2} &= q_n r_{n-1} + r_n, 0 < r_n < r_{n-1} \\ r_{n-1} &= q_{n+1} r_n + 0 \end{aligned}$$

[1] 교신저자: (712-702) 경상북도 경산시 하양읍 하양로 13-13, 대구가톨릭대학교 바이오융합대학 수학과, 교수. E-mail: dhkim@cu.ac.kr
[2] (712-702) 경상북도 경산시 하양읍 하양로 13-13, 대구가톨릭대학교 사범대학 수학교육과, 교수

수학자 라메 (Gabriel Lame)는 유클리드 알고리즘에 요구되는 단계의 수는 많아야 작은 수의 자릿수의 5배임을 증명하였다 (Knuth, 1998). 실제로 유클리드 알고리즘의 단계 수는 근사적으로 $\frac{12\log 2}{\pi^2}\log b$임을 Dixon (1970)이 밝힌 바 있다.

확장 유클리드 알고리즘은 두 정수 a, b $(a > b)$에 대하여 최대공약수 g는 물론이고

$$s \times a + t \times b = g$$

를 만족하는 두 정수 s와 t를 구하는 알고리즘이다. 확장 유클리드 알고리즘의 단계수는 유클리드 알고리즘이 필요로 하는 단계수와 동일하다. 그러나 각 단계에서 사용하는 계산에서는 좀 더 많은 계산을 진행한다. 확장 유클리드 알고리즘의 결과로 부터 우리는 쉽게 주어진 수의 곱셈에 대한 역원 (multiplicative inverse)을 얻을 수도 있다.

최근 들어 컴퓨터를 활용하여 문제를 해결하는 컴퓨터 집약적 방법 (computer intensive method)이 여러분야에서 나타나고 있다. 특히 엑셀의 매크로 기능을 이용한 도구들이 계속 개발되고 있다. Choi와 Ha (2012)는 엑셀 매크로기능을 이용하여 베이즈 (Bayes) 정리 교육도구를 개발하였으며 Choi와 Ha (2011)는 엑셀 매크로를 이용한 절차 중심의 통계교육도구를 개발하였다. 또 Choi와 Kim (2010)은 엑셀의 함수를 이용한 표본추출에 관하여 연구한 바 있으며 최근 들어 Kim (2012)은 암호학의 영역에서 DES의 라운드 키 생성 엑셀 매크로를 개발한 바 있다. 본 논문의 2절에서는 유클리드 알고리즘과 확장 유클리드 알고리즘을 소개하고 이들을 컴퓨터 집약적 방법으로 해결하기 위한 프로세스 (컴퓨터 알고리즘)를 설명하였다. 3절에서는 이들 알고리즘들을 바탕으로 실제로 컴퓨터를 이용하여 해결하는 방법으로서 C 언어를 이용하는 방법과 마이크로 소프트 엑셀을 이용하는 두 가지 방법을 설명하고 소스프로그램과 엑셀 매크로 결과를 포함하였다. 마지막으로 4절에서는 결론을 나타내었다.

2. 확장 유클리드 알고리즘의 프로세스

2.1. 유클리드 알고리즘

1절에서도 설명하였듯이 유클리드 알고리즘은 여러가지 면에서 효율적 결과를 제공한다. 유클리드 알고리즘을 이용하여 아주 큰 수 (예로서 500자리의 수) 들에 대한 최대공약수를 구하는 문제는 실제로 공개키 암호화에서 빈번하게 일어난다. 공개키 암호 중 RSA 암호에서는 안전을 위하여 사용하는 두 소수의 크기를 512비트로 정하고 있다. 512비트는 십진수로 약 154자리 수에 해당한다. 현실적으로는 전혀 사용되지 않는 수처럼 여겨지지만 실제로는 이 보다 더 큰 수도 자주 사용되고 있다. 이런 이유로 유클리드 알고리즘의 컴퓨터 집약적 방법의 필요성이 대두되었다. Figure 2.1은 유클리드 알고리즘의 과정을 나타내고 있다.

이 프로세스의 핵심은 나머지 r이 0이 될 때가지 계속 수행한다는 점이다. 이 때에 최대공약수는 r_1이 된다. 이 과정을 따라서 계산한 결과는 Table 2.1과 같다. 이 경우는 세번의 유클리드 알고리즘 단계를 거쳐 최대공약수가 구해졌음을 알 수 있다.

Table 2.1 Calcualtion of Euclidean algorithm

q	r_1	r_2	r
1	323	247	76
3	247	76	19
4	79	19	0
	19	0	

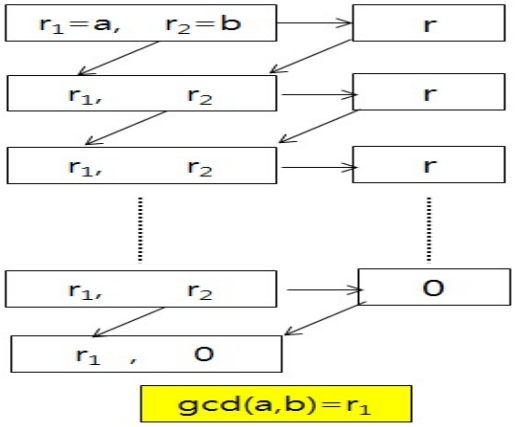


Figure 2.1 Euclidean algorithm processing

2.2. 확장 유클리드 알고리즘과 곱셈에 대한 역원

확장 유클리드 알고리즘은 말 그대로 유클리드 알고리즘을 확장시켜 놓은 것이다. 확장 유클리드 알고리즘은 최대공약수를 구하는 과정 그 자체를 확장시켜 주어진 두 정수 자리에 1과 0 그리고 0과 1을 차례로 대입하여 알고리즘을 반복하여 원하는 값을 구하는 방법이다. Figure 2.2에 확장 유클리드 알고리즘의 프로세스가 나타나 있다.

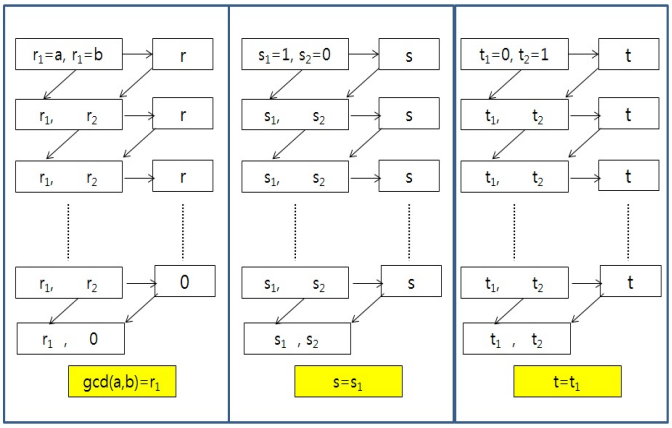


Figure 2.2 Extended Eculidean algorithm processing

확장 유클리드 알고리즘은 유클리드 알고리즘의 각 단계에서 한 쌍 대신 세 쌍의 계산과 교환을 진행하는 것이 특징이다. 여기서 사용되는 변수는 r을 포함하여 s와 t를 사용한다. 여기서 주의할 점은 변수 r, s, t의 계산은 유사하지만 각각의 영역내에서 s와 t는 나머지의 값에 제약이 없다는 것과 주어진 두 정수 r_1과 r_2의 몫 q가 다른 두 연산에 사용된다는 점이다. 이 프로세스를 따라서 계산한 결과가 Table 2.2에 나타나 있다.

Table 2.2 Calculation of Extended euclidean algorithm

q	r_1	r_2	r	s_1	s_2	s	t_1	t_2	t
5	164	28	21	1	0	1	0	1	-5
1	28	21	7	0	1	-1	1	-5	6
3	21	7	0	1	-1	4	-5	6	23
	7	0		-1	4		6	-23	

이제 곱셈에 대한 역원을 구하는 과정을 고려하자. 두 정수 a와 n에 대하여 a를 n으로 나눈 나머지에만 관심을 가지는 연산을 모듈로 연산 (modulo operator)이라 부르며 계산된 나머지 r을

$$r \equiv a \mod n$$

로 나타낸다. 모듈로 n을 이용하는 모듈로 연산의 결과는 항상 0과 $n-1$사이의 정수로 나타난다. 모듈로 n 연산의 결과는 하나의 집합을 형성하게 되는데 이 집합 Z_n을 잉여류 (set of residue)라 부른다. 두 정수가 합동 (congruence)이라 함은 모듈로 n에 대하여 나머지가 같음을 의미하고 합동기호 $\equiv$를 이용하여 표현한다. 예로서 $2 \equiv 12 \mod 10$이다. 잉여류 집합 Z_n에서의 연산을 생각하여 보자. Z_n에서는 덧셈, 뺄셈, 곱셈의 연신이 정의될 수 있다. 연산의 결과에는 항상 모듈로 n이 작동하게 되어 그 결과는 Z_n의 원소가 된다. 그러나 곱셈에 대한 역원의 경우는 그렇지 않을 수 있다. 곱셈에 대한 역원이란 잉여류 집합 Z_n에 속하는 두 수 a, b 중

$$a \times b \equiv 1 \mod n$$

을 만족하는 b를 a의 곱셈에 대한 역원 (역원, multiplicative inverse)이라고 부른다. 예로서 모듈로가 10인 경우 3의 역원은 7이다. 모듈로 n 연산에서 역원은 존재할 수도 있고 존재하지 않을 수도 있다. 어떤 정수 a가 Z_n에서 곱셈에 대한 역원을 가지기 위해서는 a와 n은 서로 소 (relatively prime)의 관계를 가져야 한다. Figure 2.3은 곱셈에 대한 역원을 구하는 프로세스를 나타내고 있다.

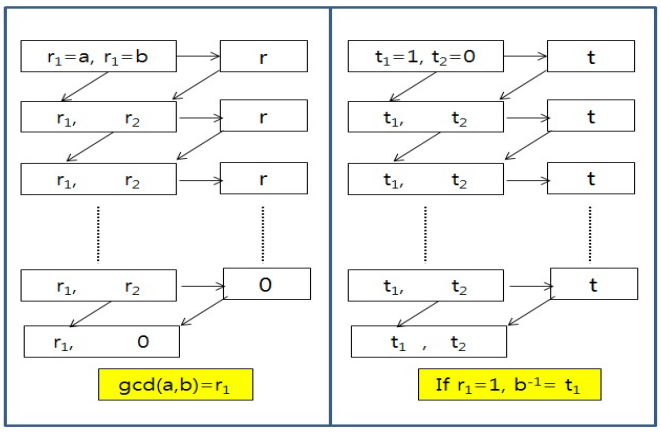


Figure 2.3 Multiplicative inverse processing

곱셈에 대한 역원을 구하는 과정은 확장 유클리드 알고리즘의 부분적 활용을 고려하면 된다.

$$s \times a + t \times b = g$$

로 부터 s를 0으로 두면 자연스럽게

$$t \times b = g$$

를 얻게된다. 주어진 수 b와 모듈로 n이 서로 소일 경우에 역원이 존재함을 이용하면 즉 b와 n의 초대공약수가 1임을 적용하여

$$t \times b = 1$$

을 얻게 된다. 즉 t가 구하고자 하는 b에 대하여 곱셈에 대한 역원이 된다. Table 2.3은 Z_{26}일 경우 곱셈에 대한 역원을 구하는 프로세스를 적용하여 얻은 결과이다.

Table 2.3 Calculation of multiplicative inverse

q	r_1	r_2	r	t_1	t_2	t
2	26	11	4	0	1	-2
2	11	4	3	1	-2	5
1	4	3	1	-2	5	-7
3	3	1	0	5	-7	26
	1	0		-7	26	

Table 2.3에서 t_1열에 해당되는 -7이 바로 모듈로 26에서 11의 역원이다. 이는 $-7 \mod 26 \equiv 19 \mod 26$으로 부터 11의 곱셈에 대한 역원이 19임을 확인할 수 있다.

3. 확장 유클리드 알고리즘 컴퓨터 집약적 방법

2절에서 설명한 유클리드 알고리즘이나 확장 유클리드 알고리즘은 아주 큰 수의 경우에 유용하지만 실제로 이 알고리즘들을 컴퓨터에서 활용하는 것이 훨씬 더 중요한 일이다. 확장 유클리드 알고리즘의 소스 프로그램은 그 내용상의 쉬운 프로세스 전개로 인하여 일반적으로 공개되어 있지 않다. 확장 유클리드 알고리즘을 다루는 대부분의 학생들은 손으로 몇번의 계산과정을 확인하면서 익히는 경우가 대부분이다. 실제로 아주 큰 수들의 최대공약수나 곱셈에 대한 역원, 나아가 소인수 분해 등의 경우에 봉착할 경우 컴퓨터 집약적 방법에 의존하지 않을 수 없다. 본 절에서는 확장 유클리드 알고리즘의 컴퓨터 집약적 방법의 일환으로서 C 프로그래밍 방법과 마이크로소프트 엑셀을 이용하는 두 가지 방법을 활용하고자 한다.

C 프로그래밍에 익숙한 사용자들은 본 절에서 소개되는 C 소스코드를 활용하여 확장 유클리드 알고리즘을 해결하면 되고 C 프로그래밍에 익숙하지 않거나 마이크로소프트 엑셀 프로그램을 선호하는 사용자들은 본 논문에서 제안하는 엑셀 매크로를 활용하여 해결할 수 있다. Figure 3.1 부터 Figure 3.2 까지는 확장 유클리드 알고리즘에 대한 C 소스프로그램과 결과를 나타내고 있으며 Figure 3.3은 마이크로소프트 엑셀을 이용하는 확장 유클리드 알고리즘의 매크로 결과를 보여주고 있다.

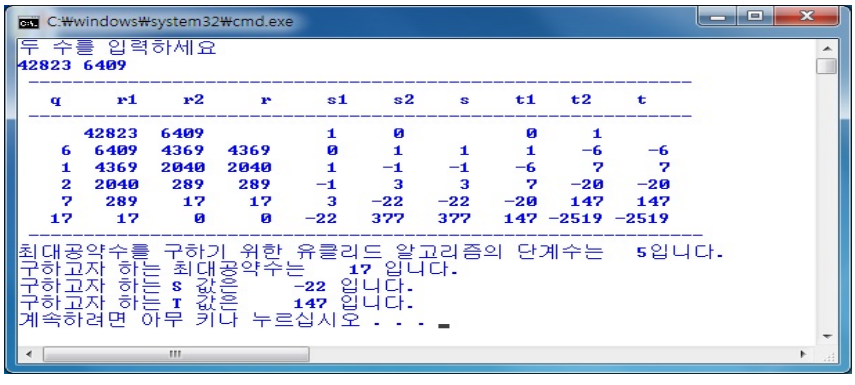


Figure 3.1 Output for extended Euclidean C source

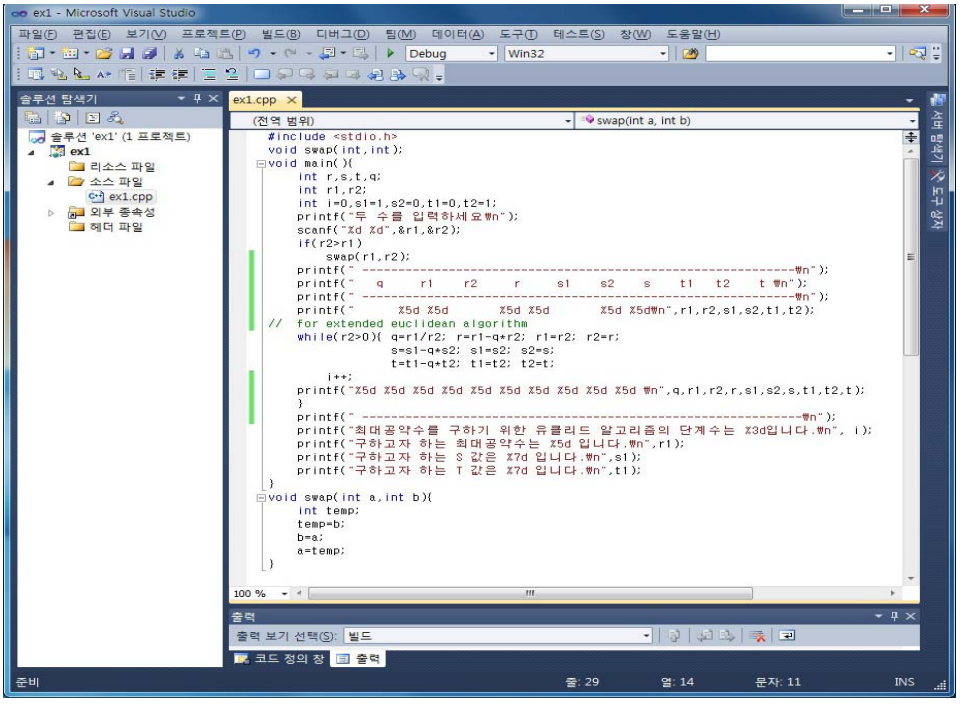
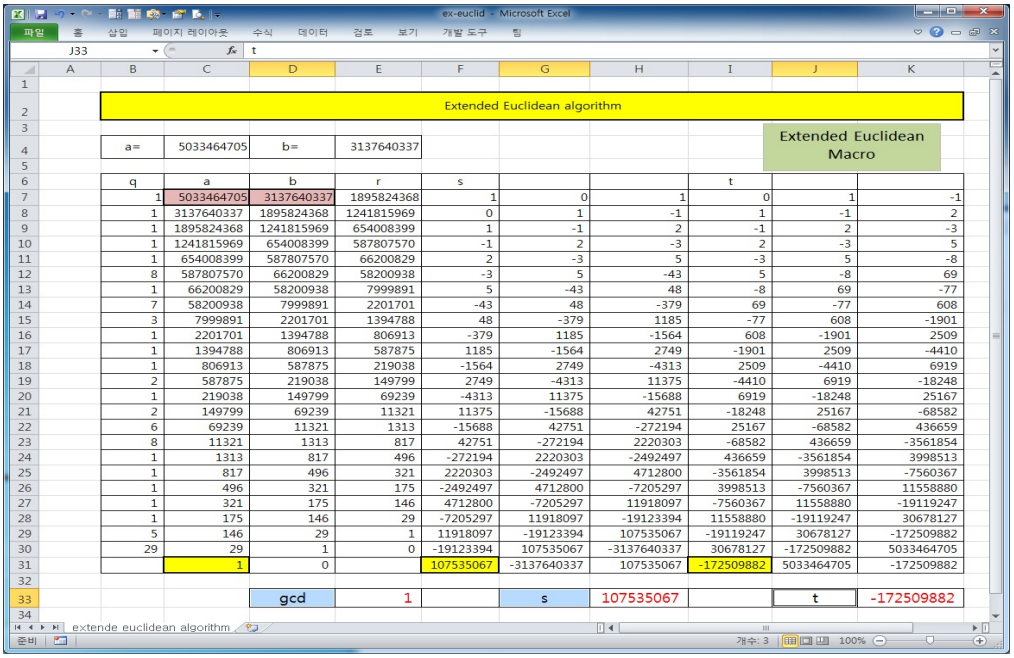


Figure 3.2 C source for extended Euclidean algorithm



q	a	b	r	s			t		
1	5033464705	3137640337	1895824368	1	0	1	0	1	-1
1	3137640337	1895824368	1241815969	0	1	-1	1	-1	2
1	1895824368	1241815969	654008399	1	-1	2	-1	2	-3
1	1241815969	654008399	587807570	-1	2	-3	2	-3	5
1	654008399	587807570	66200829	2	-3	5	-3	5	-8
8	587807570	66200829	58200938	-3	5	-43	5	-8	69
1	66200829	58200938	7999891	5	-43	48	-8	69	-77
7	58200938	7999891	2201701	-43	48	-379	69	-77	608
3	7999891	2201701	1394788	48	-379	1185	-77	608	-1901
1	2201701	1394788	806913	-379	1185	-1564	608	-1901	2509
1	1394788	806913	587875	1185	-1564	2749	-1901	2509	-4410
1	806913	587875	219038	-1564	2749	-4313	2509	-4410	6919
2	587875	219038	149799	2749	-4313	11375	-4410	6919	-18248
1	219038	149799	69239	-4313	11375	-15688	6919	-18248	25167
2	149799	69239	11321	11375	-15688	42751	-18248	25167	-68582
6	69239	11321	1313	-15688	42751	-272194	25167	-68582	436659
8	11321	1313	817	42751	-272194	2220303	-68582	436659	-3561854
1	1313	817	496	-272194	2220303	-2492497	436659	-3561854	3998513
1	817	496	321	2220303	-2492497	4712800	-3561854	3998513	-7560367
1	496	321	175	-2492497	4712800	-7205297	3998513	-7560367	11558880
1	321	175	146	4712800	-7205297	11918097	-7560367	11558880	-19119247
1	175	146	29	-7205297	11918097	-19123394	11558880	-19119247	30678127
5	146	29	1	11918097	-19123394	107535067	-19119247	30678127	-172509882
29	29	1	0	-19123394	107535067	-3137640337	30678127	-172509882	5033464705
	1	0		107535067	-3137640337	107535067	-172509882	5033464705	-172509882

Figure 3.3 Extended Euclidean algorithm Excel macro

실제로 RSA (Rivest 등, 1978) 공개키 암호 시스템은 현재 사용되고 있는 암호화 방법으로서 최대공약수와 확장 유클리드 알고리즘을 활용하고 있다. 512비트 이상의 두 소수 p와 q를 선택하고 두 수를 곱하여 합성수 $n = p \times q$를 구한 후 오일러 함수 $\phi(n) = (p-1) \times (q-1)$을 구한다. 이제 잉여류 집합 $Z_{\phi(n)}$에서 서로 소인 두 수 d와 e를 선택하여 그 중 하나인 e와 합성수 n을 공개키 값으로 사용한다. 이 때 모듈로 연산의 특징 상

$$d \times e \equiv 1 \mod \phi(n)$$

임은 자명하다. 이제 평문 숫자 p는

$$c = p^e \mod n$$

으로 암호화되어 보내지고 이 암호 숫자 c는

$$p = c^d \mod n$$

에 의해 복호화되는 암호시스템이 RSA 공개키 암호시스템이다. 이 복호화 과정에서 비밀키 값인 d를 구하기 위하여

$$d = e^{-1} \mod \phi(n)$$

을 구할 때에 즉 모듈로 $\phi(n)$에서 e에 대한 곱셈의 역원 d을 구할 때에 확장 유클리드 알고리즘이 사용된다. 여기서 기억하여야 할 사실 중의 하나는 우리가 다루는 수가 상당히 큰 수라는 점이다. 또한 암호화나 복호화 과정에서 멱승 연산이 요구된다는 점이다.

4. 결론 및 토의

본 연구에서는 유클리드 알고리즘과 확장 유클리드 알고리즘 그리고 모듈로 연산에서의 곱셈에 대한 역원 등에 대하여 조사하고 이들을 컴퓨터를 이용하여 구현하는 두 가지 컴퓨터 집약적 방법을 제안하였다. 제안된 두 가지 컴퓨터 집약적인 방법은 각각 C 프로그램과 마이크로소프트사의 엑셀 프로그램을 이용하여 구현하였다. 또한 현대 사회에서 실제로 사용되고 있는 공개키 암호 시스템인 RSA에서 확장 유클리드 알고리즘의 활용도 첨언하였다. 제안된 두 가지 컴퓨터 집약적 방법 모두 저 마다의 장점을 지니고 있지만 사용자들의 상황에 따라서 쉽게 그리고 간단하게 적용될 수 있는 방법을 모색하여 활용하면 편리할 것이다. 강력한 C나 C++ 프로그래밍을 이용하여 아주 큰 수들에 대한 확장 유클리드 알고리즘의 계산과 RSA 암호화를 구현할 수도 있고 때로는 프로그래밍의 부담을 제거하고 대부분의 컴퓨터에 기본적으로 탑재되어 있는 마이크로소프트 엑셀을 이용하여 확장 유클리드 알고리즘을 해결하는 방법도 교육적으로 상당한 의의가 있다고 사료된다.

References

Choi, H. S. and Ha, J. (2011). Development of process-oriented education tool for Statistics with Excel Macro. *Journal of the Korean Data & Information Science Society*, **22**, 643-650.

Choi, H. S. and Ha, J. (2012). Development of Bayes' rule education tool with Excel Macro. *Journal of the Korean Data & Information Science Society*, **23**, 905-912.

Choi, H. S. and Kim, T. Y. (2010). A study on sampling using the function of excel. *Journal of the Korean Data & Information Science Society*, **21**, 481-491.

Dixon, J. (1970). The number of steps in the Euclidean algorithm. *Journal of Number Theory*, **2**, 414-422.

Kim, D. (2012). On the development of DES round key generator based on Excel Macro. *Journal of the Korean Data & Information Science Society*, **23**, 1203-1212.

Knuth, D. E. (1998). *The art of computer programming, volume 2: Seminumerical algorithms*, Addison-Wesley Professional, Boston, MA.

Rivest, R. L., Shamir, A. and Adleman, L. (1978). A method of obtaining digital signature and public-key cryptosystem. *Communication of the Association for Computing Machinery*, **21**, 120-126.

Journal of the Korean Data & Information Science Society
2014, **25**(6), 1467–1474

http://dx.doi.org/10.7465/jkdi.2014.25.6.1467
한국데이터정보과학회지

Computer intensive method for extended Euclidean algorithm

Daehak Kim[1] · Kwang Sik Oh[2]

[1]Department of Mathematics, Catholic University of Daegu
[2]Department of Mathematics Education, Catholic University of Daegu

Received 9 October 2014, revised 6 November 2014, accepted 11 November 2014

Abstract

In this paper, we consider the two computer intensive methods for extended Euclidean algdrithm. Two methods we propose are C-programming based approach and Microsoft excel based method, respectively. Thses methods are applied to the derivation of greatest commnon devisor, multiplicative inverse for modular operation and the solution of diophantine equation. Concrete investigation for extended Euclidean algorithm with the computer intensive process is given. For the application of extended Euclidean algorithm, we consider the RSA encrytion method which is still popular recently.

Keywords: Computer intensive method, extende Euclidean algorithm, greatest common devisor, modular operation, multiplicative inverse.

[1] Corresponding author: Professor, Department of Mathematics, Catholic University of Daegu, Kyungsan 712-702, Korea. E-mail: dhkim@cu.ac.kr
[2] Professor, Department of Mathematics Education, Catholic University of Daegu, Kyungsan 712-702, Korea.