The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 14, No. 2, pp.121-126, Apr. 30, 2014. pISSN 2289-0238, eISSN 2289-0246

http://dx.doi.org/10.7236/JIIBC.2014.14.2.121
JIIBC 2014-2-17

십진수의 자리이동-덧셈 곱셈법

Shift-and-Add Multiplication Algorithm for Decimal System

이상운*

Sang-Un Lee*

요 약 큰 -자리수의 2개 10진수에 대한 곱셈을 보다 빠르게 수행하는 방법은 존재하는가? 이 문제는 수학과 컴퓨터공학 분야에서 미해결 문제로 남아 있다. 이 문제에 대해 곱셈 횟수를 줄이는 연구로는 Karatsuba와 Toom-Kook 알고리즘이 있다. 본 논문은 곱셈 횟수를 줄이는 방법과는 완전히 별개로, 10진수 곱셈을 전적으로 덧셈만으로 효율적으로 수행하는 방법을 제안하였다. 제안된 방법은 2진수의 자리이동-덧셈법만으로도 RSA-100과 같이 컴퓨터로 수행이 불가한 매우 큰 자리수의 10진수 곱셈을 수행할 수 있음을 보였다. 제안된 방법은 수행 복잡도 (n)의 덧셈으로 곱셈을 수행한다.

Abstract The problem of finding the fastest algorithm for multiplication of two large n-digit decimal numbers remains unsolved in the field of mathematics and computer science. To this problem so far two algorithms - Karatsuba and Toom-kook - have been proposed to shorten the number of multiplication. In the complete opposite of shorten the number of multiplication method, this paper therefore proposes an efficient multiplication algorithm using additions completely. The proposed algorithm totally applies shift-and-add algorithm of binary system to large digits of decimal number multiplication for example of RSA-100 this problem can't perform using computer. This algorithm performs multiplication purely with additions of complexity of $O(n^2)$.

Key Words : Multiplication, Long (grade-school) multiplication, Shift-and-add, Karatsuba multiplication, Toom-kook multiplication

Ⅰ. 서 론

2개의 자리수 10진 숫자에 대해 곱셈을 기존에 제안된 알고리즘 보다 더 빨리 수행할 수 있는 방법은 존재하는가? 이 문제는 컴퓨터과학 분야에서 미해결 문제 중 하나이다.[1] 이 문제는 $RSA-100_{(10)}$, $RSA-1024_{(2)}(309_{(10)})$ $RSA-2048_{(2)}(617_{(10)})$과 같은 암호를 생성할 때 많이 사용된다.

만약, $p\times q=r$으로 $c=100(RSA-100)$을 생성할 경우 $n=l(p)=l(q)=50$의 십진수 자리수이다. 이 경우 가장 일반적으로 알려진 초등수학 곱셈법 (grade-school or long multiplication)을 적용하면 n^2회의 곱셈을 수행해야 하므로 수행 복잡도는$O(n^2)$이다.

이러한 초등수학 곱셈법의 곱셈 수행횟수를 줄이기 위한 방법으로 Karatsuba와 Toom-Kook 알고리즘이 제안되었다.[2-6] Karatsuba 알고리즘[3]은 주어진 수를 양분

*정회원, 강릉원주대학교 과학기술대학 멀티미디어공학과
접수일자 : 2013년 12월 24일, 수정완료 : 2014년 3월 10일
게재확정일자 : 2014년 4월 11일

Received: 24 December, 2013 / Revised: 10 March, 2014 / Accepted: 11 April, 2014
*Corresponding Author: sulee@gwnu.ac.kr
Dept. of Multimedia Eng., Gangneung-Wonju National University, Korea

한 1차 다항식으로 표현하여 일반적인 $/2\times n/2$ 곱셈 4회, 덧셈 3회를 $n/2\times n/2$ 곱셈 3회, 덧셈 (뺄셈 포함) 6회로, 곱셈을 1회 줄이는 대신 덧셈을 3회 증가시키는 방법을 제안하였다.

Karatsuba 곱셈법은 복잡도가 $(n^{\log})=O(n^{1.585})$이다. Toom-Kook의 Toom-3법[4]은 주어진 수를 3등분한 2차 다항식으로 표현하여 Karatsuba 곱셈법의 $n/3\times n/3$의 곱셈 6회, 덧셈 (뺄셈 포함) 17회를 $n/3\times n/3$의 곱셈 5회, $n/3\times 1$의 곱셈 6회와 덧셈 5회를 증가시키는 방법을 제안하였다.

본 논문에서는 곱셈을 전혀 하지 않고 완전히 덧셈만으로 곱셈을 수행하는 알고리즘을 제안한다. 2장에서는 Karatsuba와 Toom-3 법을 고찰해 본다. 3장에서는 완전히 덧셈만으로 곱셈을 수행하는 자리이동-덧셈의 곱셈법을 제안한다. 4장에서는 제안된 곱셈법을 적용하여 본다.

Ⅱ. 관련연구

초등수학 곱셈법을 보다 효율적으로 수행하기 위한 연구로는 곱셈 횟수 감소와 하드웨어 곱셈기 구현으로 연구가 진행되고 있다. 본 장에서는 곱셈횟수 감소법인 Karatsuba와 Toom-3 알고리즘을 고찰한다.

Karatsuba 곱셈법[3]은 분할정복법 (divide-and-conquer), 이진 분할법 (binary splitting), 또는 양분 원칙 (dichotomy principle)이라고도 부른다. 이 방법은 2개의 수 p,q에 대해 다음과 같이 양분하여 곱셈을 수행한다.

1차 다항식에 대한 일반적인 방법은 식 (1)과 같이 곱셈 4회와 덧셈 3회를 수행한다.

p \ q	n_1x	n_0
m_1x	$m_1n_1x^2$	m_1n_0x
m_0	m_0n_1x	m_0n_0

$$
\begin{aligned}
&p(x)=m_1x+m_0,\ q(x)=n_1x+n_0\\
&r(x)=p(x)q(x)=r_2x^2+r_1x+r_0 \qquad (1)\\
&\quad =m_1n_1x^2+(m_1n_0+m_0n_1)x+m_0n_0
\end{aligned}
$$

Karatsuba 곱셈법은 식 (1)에서 $r_1=(m_1n_0+m_0n_1)$에 대해 식 (2) 또는 식 (3)으로 변형시켜 곱셈 3회와 덧셈 (뺄셈 포함) 6회를 수행하는 방법으로 변형시켜 곱셈을 1회 감소시켰다. 즉, 곱셈 1회를 감소시킬 경우 덧셈 (뺄셈 포함)은 3회가 증가한다.

$$
\begin{aligned}
&(m_1+m_0)(n_1+n_0)=m_1n_1+m_1n_0+m_0n_1+m_0n_0\\
&r_1=m_1n_0+m_0n_1\\
&\quad =(m_1+m_0)(n_1+n_0)-(m_1n_1+m_0n_0)\\
&r(x)=m_1n_1x^2\\
&\qquad +[(m_1+m_0)(n_1+n_0)-(m_1n_1+m_0n_0)]x \qquad (2)\\
&\qquad +m_0n_0\\
&(m_1-m_0)(n_1-n_0)=m_1n_1-(m_1n_0+m_0n_1)+m_0n_0\\
&r_1=m_1n_0+m_0n_1\\
&\quad =(m_1n_1+m_0n_0)-(m_1-m_0)(n_1-n_0)\\
&r(x)=m_1n_1x^2\\
&\qquad +[(m_1n_1+m_0n_0)-(m_1-m_0)(n_1-n_0)]x \qquad (3)\\
&\qquad +m_0n_0
\end{aligned}
$$

Karatsuba 곱셈법을 Toom-3[4]와 비교하기 위해 3등분하여 2차 다항식으로 확장시켜 보자. 일반적인 방법은 식 (4)와 같이 곱셈 9회와 덧셈 (뺄셈 포함) 8회를 수행한다.

p \ q	n_2x^2	n_1x	n_0
m_2x^2	$m_2n_2x^4$	$m_2n_1x^3$	$m_2n_0x^2$
m_1x	$m_1n_2x^3$	$m_1n_1x^2$	m_1n_0x
m_0	$m_0n_2x^2$	m_0n_1x	m_0n_0

$$
\begin{aligned}
&p(x)=m_2x^2+m_1x+m_0,\ q(x)=n_2x^2+n_1x+n_0\\
&r(x)=p(x)q(x)=r_4x^4+r_3x^3+r_2x^2+r_1x+r_0 \qquad (4)\\
&\quad =m_2n_2x^4\\
&\qquad +(m_1n_2+m_2n_1)x^3\\
&\qquad +(m_0n_2+m_1n_1+m_2n_0)x^2\\
&\qquad +(m_0n_1+m_1n_0)]x\\
&\qquad +m_0n_0
\end{aligned}
$$

이 경우, Karatsuba 곱셈법은 식 (5)나 식 (6)과 같이 $n/3\times n/3$의 곱셈 6회와 덧셈 (뺄셈 포함) 17회를 수행한다. 즉, 일반적인 방법에 비해 곱셈 3회를 감소시킨 결과 $3\times 3=9$회의 덧셈이 증가한다.

$$
\begin{aligned}
&(m_2+m_1)(n_2+n_1)=m_2n_2+m_2n_1+m_1n_2+m_1n_1\\
&r_3=m_1n_2+m_2n_1\\
&\quad =(m_2+m_1)(n_2+n_1)-(m_2n_2+m_1n_1)\\
&(m_2+m_0)(n_2+n_0)=m_2n_2+m_2n_0+m_0n_2+m_0n_0
\end{aligned}
$$

$$= m_0 n_2 + m_1 n_1 + m_2 n_0$$
$$= (m_2 + m_0)(n_2 + n_0) - (m_2 n_2 + m_0 n_0) + m_1 n_1$$

$$(m_1 + m_0)(n_1 + n_0) = m_1 n_1 + m_1 n_0 + m_0 n_1 + m_0 n_0$$

$$r_1 = m_1 n_0 + m_0 n_0$$
$$= (m_1 + m_0)(n_1 + n_0) - (m_1 n_1 + m_0 n_0)$$

$$\begin{aligned}(x) = m\ n_2 x^4 \\ + [(m_2 + m_1)(n_2 + n_1) - (m_2 n_2 + m_1 n_1)]x^3 \\ + [(m_2 + m_0)(n_2 + n_0) - (m_2 n_2 + m_0 n_0) + m_1 n_1]x^2 \\ + [(m_1 + m_0)(n_1 + n_0) - (m_1 n_1 + m_0 n_0)]x \\ + m_0 n_0\end{aligned} \quad (5)$$

$$\begin{aligned}r(x) = m_2 n_2 x^4 \\ + [(m_2 n_2 + m_1 n_1) - (m_2 - m_1)(n_2 - n_1)]x^3 \\ + [(m_2 n_2 + m_1 n_1 + m_0 n_0) - (m_2 - m_0)(n_2 - n_0)]x^2 \\ + [(m_1 n_1 + m_0 n_0) - (m_1 - m_0)(n_1 - n_0)]x \\ + m_0 n_0\end{aligned} \quad (6)$$

Toom-Cook 곱셈법의 Toom-3는 주어진 수를 3등분한 2차 다항식에 대해 다음과 같이 수행되며, $n/3 \times n/3$의 곱셈은 5회, $n/3 \times 1$의 곱셈 (나눗셈 포함)은 6회, 덧셈 (뺄셈 포함)은 22회를 수행한다. 이 방법은 Karatsuba 곱셈법의 $n/3 \times n/3$ 곱셈 횟수를 1회 감소시키는 대신 $n/3 \times 1$의 곱셈 (나눗셈 포함)은 6회 증가하였으며, 덧셈 (뺄셈 포함)은 5회 증가하였다.

$$p(x) = p_2 x^2 + p_1 x + p_0, \quad (x) = q\ x^2 + q_1 x + q_0$$
$$r(x) = p(x)q(x) = r_4 x^4 + r_3 x^3 + r_2 x^2 + r_1 x + r_0$$

$$p(0) = m_0,\ q(0) = n_0$$
$$p(\infty) = m_2,\ q(\infty) = n_2$$
$$p_0 = m_2 + m_0,\ q_0 = n_2 + n_0$$
$$p(1) = m_2 + m_1 + m_0 = p_0 + m_1$$
$$q(1) = n_2 + n_1 + n_0 = q_0 + n_1$$
$$p(-1) = m_2 - m_1 + m_0 = p_0 - m_1$$
$$q(-1) = n_2 - n_1 + n_0 = q_0 - n_1$$
$$p(-2) = 4m_2 - 2m_1 + m_0 = 2[p(-1) + m_2] - m_0$$
$$q(-2) = 4n_2 - 2n_1 + n_0 = 2[q(-1) + n_2] - n_0$$

$$r(0) = p(0) \times q(0) = m_0 n_0$$
$$r(1) = p(1) \times q(1) = r_4 + r_3 + r_2 + r_1 + r_0$$
$$r(-1) = p(-1) \times q(-1) = r_4 - r_3 + r_2 - r_1 + r_0$$
$$r(-2) = p(-2) \times q(-2) = 16r_4 - 8r_3 + 4r_2 - 2r_1 + n_0)$$
$$r(\infty) = p(\infty) \times q(\infty) = m_2 n_2$$

$$r_0 \leftarrow r(0) = m_0 n_0$$
$$r_4 \leftarrow r(\infty) = m_2 n_2$$
$$a \leftarrow [r(-2) - r(1)]/3 = 5r_4 - 3r_2 + r_2 - r_1$$
$$b \leftarrow [r(1) - r(-1)]/2 = r_3 + r_1$$
$$c \leftarrow r(-1) - r(0) = r_4 - r_3 + r_2 - r_1$$
$$r_3 \leftarrow (c - a)/2 + 2r_4$$
$$r_2 \leftarrow (c + b) - r_4$$
$$r_1 \leftarrow b - r_3$$
$$r(x) = p(x)q(x) = r_4 x^4 + r_3 x^3 + r_2 x^2 + r_1 x + r_0$$

Karatsuba와 Toom-3 곱셈법 모두 단지 곱셈 횟수를 1회 감소시키는 연구를 수행하였다. 그럼에도 불구하고 많은 곱셈 횟수를 수행해야 하는 어려움이 있다. 곱셈은 덧셈에 비해 많은 시간을 소요한다. 만약, 하드웨어 곱셈기 도움 없이 소프트웨어적으로 완전히 덧셈만으로 곱셈을 수행할 수 있다면 가장 이상적인 곱셈법이 될 것이며, 곱셈에 소요되는 시간도 크게 단축시킬 수 있을 것이다. 따라서, 3장에서는 완전히 덧셈만으로 곱셈을 수행하는 방법을 제안한다.

Ⅲ. 자리이동-덧셈 곱셈법

본 장에서 제안하는 곱셈법은 곱셈을 전혀 수행하지 않고, 완전히 덧셈만으로 곱셈을 수행하는 방법이다. 이로 인해, 불필요하게 고가의 하드웨어 곱셈기를 이용하지 않아도 되는 장점을 갖고 있다. 제안된 방법은 2진수의 곱셈법인 자리이동-덧셈법 (shift-and-add)에 기반하여 10진수 곱셈법을 제안한다.

$c = a \times b$를 자리이동-덧셈법으로 계산하여 보자. 여기서 a,b에 있는 숫자들을 x라 하자. 이 경우 a,b 중에서 자리수가 보다 작거나 0이 보다 많거나 $_{\max}x - _{\min}x$가 보다 작은 수를 b로 설정한다.

$a = 123456, b = 654321$인 경우 $b = 6a, 5a, 4a, 3a, 2a, 1a$이며, $_{\min}x = 1, _{\max}x = 6$이다. 따라서 $a, 2a = a + a, 3a = 2a + a, 4a = 3a + a, 5a = 4a + a, 6a = 5a + a$순서로 덧셈을 계산하면서 각 xa에 대해 1의 자리까지 좌측으로 자리이동시켜 $c = c + xa$를 계산하면 된다. 여기서 $l(c) = 2l(a)$, $l(xa) = l(a) + 1$이다. 즉, 초기치는 $c = 000000654321$이며,

$\rightarrow 2a \rightarrow 3a \rightarrow \cdots \rightarrow 5a \rightarrow 6a$의 5회 덧셈과 $c = c + xa$의 자리이동 4회 덧셈으로 총 9회 덧셈을 수행하였다. 이는 그림 1에 제시되어 있다.

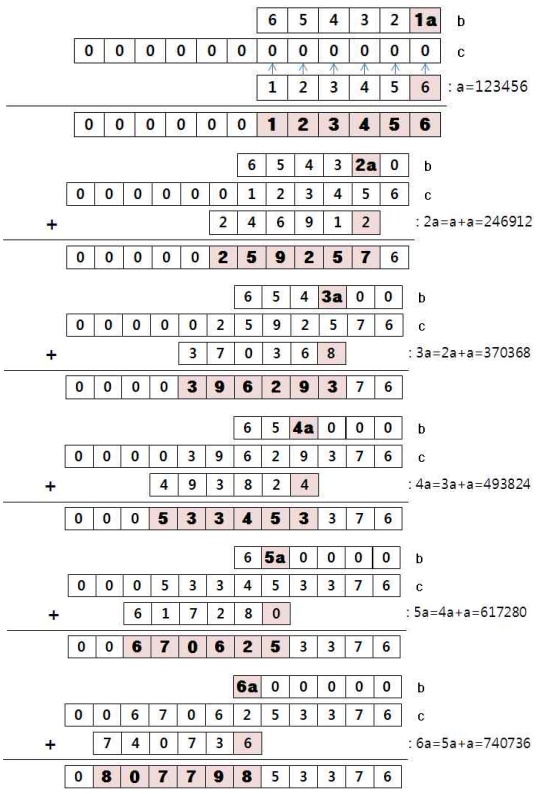


그림 1. 123456×654321의 자리이동 덧셈법
Fig. 1. Shift-and-Add Multiplication for 123456×654321

이를 Karatsuba 곱셈법으로 계산하면 그림 2와 같이 수행된다.

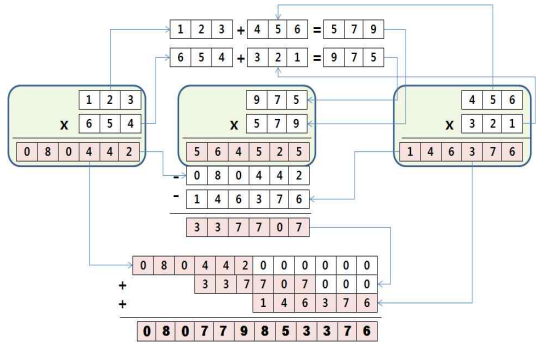


그림 2. 123456×654321의 Karatsuba 곱셈법
Fig. 2. Karatsuba Multiplication for 123456×654321

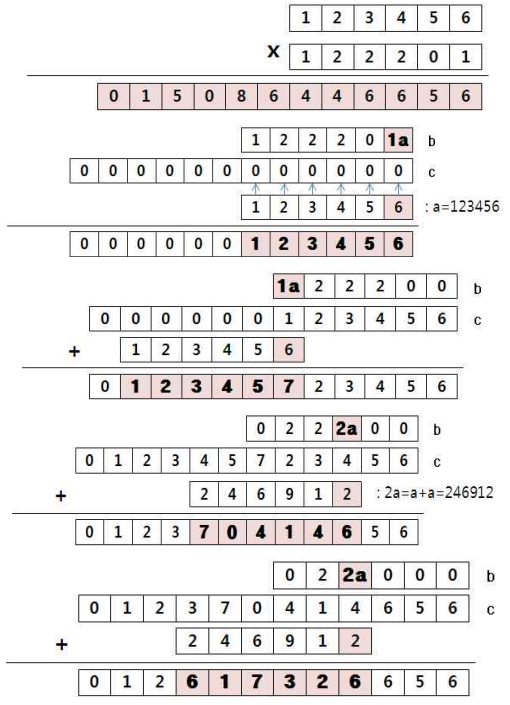
만약, ${}_{\min}x > 1$인 경우, ${}_{\min}x$까지 빠르게 도달하는 방법은 표 1과 같이 수행한다.

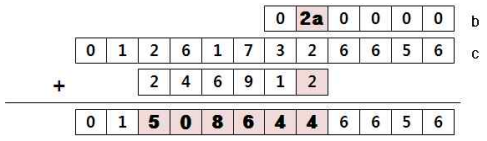
표 1. ${}_{\min}x > 1$의 빠른 계산법
Table 1. Fast Computation Method for ${}_{\min}x > 1$

${}_{\min}x$	xa			
	Step 1	Step 2	Step 3	Step 4
2	$2a = a + a$	–	–	–
3	$2a = a + a$	$3a = 2a + a$	–	–
4	$2a = a + a$	$4a = 2a + 2a$	–	–
5	$2a = a + a$	$4a = 2a + 2a$	$5a = 4a + a$	–
6	$2a = a + a$	$3a = 2a + a$	$6a = 3a + 3a$	–
7	$2a = a + a$	$3a = 2a + a$	$6a = 3a + 3a$	$7a = 6a + a$
8	$2a = a + a$	$4a = 2a + 2a$	$8a = 4a + 4a$	–
9	$2a = a + a$	$3a = 2a + a$	$8a = 4a + 4a$	$9a = 8a + a$

만약, b에서 $|1| > 1, |2| > 1, \cdots, |9| > 1$와 같이 존재하는 경우, ${}_{\min}x$까지 계산하고, ${}_{\min}x$부터 해당 숫자의 개수만큼 해당 위치에 계산하고 다음 x를 계산하는 과정을 반복 수행하면 된다.

예를 들어, $a = 123456, b = 122201$인 경우 $c = a \times b$를 자리이동-덧셈법으로 계산하여 보자. b의 ${}_{\max}x = 2$로 $a \rightarrow 2a$의 1회 덧셈과 $|1| = 2, |2| = 3$으로 $c = c + xa$의 자리이동 4회 덧셈을 수행하여 총 5회 덧셈을 하면 결과를 얻을 수 있다. 이는 그림 3에 제시되어 있다.





3. 23456×122201의 자리이동 덧셈법
Fig. 3. Shift-and-Add Multiplication for 123456×122201

제안된 방법은 다음과 같이 수행되며, $l(a)=l(b)=n$, $l(xa)=n+1, l(c)=2n$의 메모리만 있으면 완전 덧셈만으로 곱셈을 수행할 수 있는 장점을 갖고 있다.

Step 1. 주어진 수 a와 b 중에서 다음 수를 b로 설정한다.
- 자리수가 보다 작은 경우
- 0이 보다 많은 경우
- $_{\max}x-{}_{\min}x$가 보다 작은 경우

Step 2. b에서 $_{\min}x$에 대해 $a\rightarrow2a\rightarrow\cdots\rightarrow{}_{\min}xa$의 덧셈을 수행한다. b의 $_{\min}x$에 대해 1의 자리수까지 좌측으로 이동시켜 c에 저장한다. b에 있는 나머지 $_{\min}x$ 숫자 위치까지 xa의 1의 자리수를 좌측으로 이동시켜 $c=c+xa$의 덧셈을 수행한다.

Step 3. b에서 $_{\min}x+1$부터 $_{\max}x$까지 $xa=(x-1)a+a$ 덧셈을 수행하고, b의 해당 숫자가 존재하면 계속적으로 해당 숫자 위치까지 xa의 1의 자리수를 좌측으로 자리이동시켜 $c=c+xa$의 덧셈을 수행한다.

Ⅳ. 적용 결과 및 분석

본 장에서는 $a=123456789012345678901 2$, $b=98765 4321098$의 실험 데이터 1과 $SA-100, p\times q=n, l(p)=l(q)=50, l(n)=100, p=379752279369436739228088727554456278545655366381 99$, $=400946909509208810306837352927614683892148997240 61$의 실험 데이터 2에 대해 자리이동-덧셈법을 적용하여 본다.

먼저, $a=123456789012345678901 2$, $b=98765432109 8$의 실험 데이터 1의 곱셈은 $l(b)=21$인 b의 $_{\min}x=1$, $_{\max}x=9$로 $a\rightarrow2a\rightarrow\cdots\rightarrow9a$의 8회 덧셈이 수행되며, 0을 제외한 20개 숫자에 대해 $|1|=|2|=\cdots=|7|=2, |8|=|9|=3$으로 각 x에 대해 $c=c+xa$가 19회 수행된다. 따라서 총 27회의 덧셈이 된다. 이는 그림 4에 제시되어 있다.

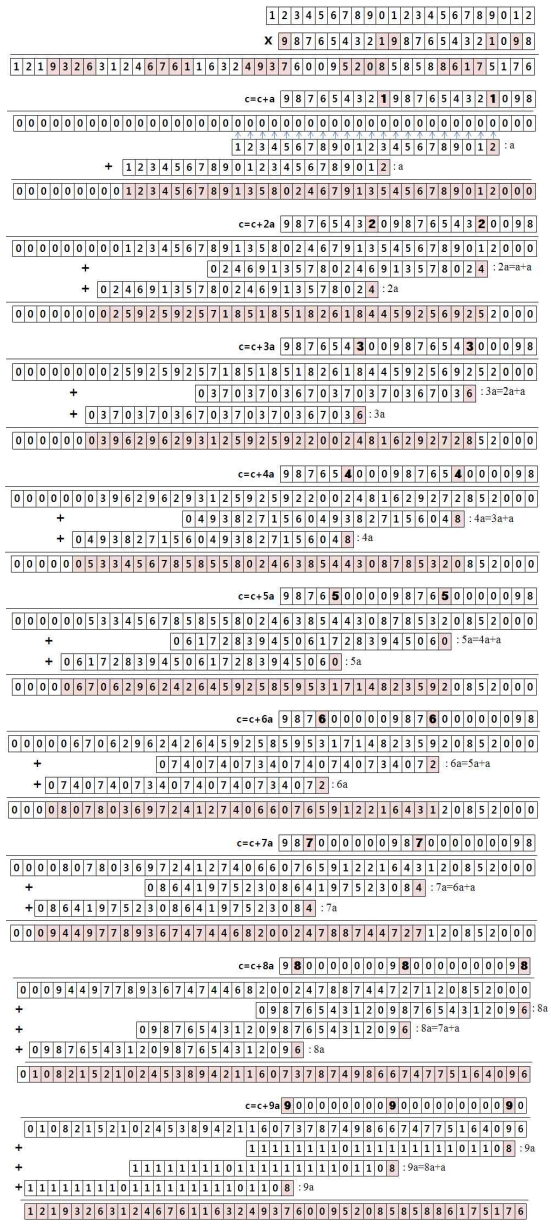


그림 4. 실험 데이터 1의 자리이동 덧셈법
Fig. 4. Shift-and-Add Multiplication for Experimental Data 1

다음으로, RSA-100의 $l(q)=50$인 q의 $|0|=8$을 제외한 42개 숫자에 대해 $_{\min}x=1$, $_{\max}x=9$로 $a\rightarrow2a\rightarrow\cdots\rightarrow9a$의 8회 덧셈이 수행되며, 각 x에 대해 $|x|$만큼 $c=c+xa$

가 계산되고 $=x+1$이 된다. 따라서, 8회+(42-1)회=49회의 덧셈으로 RSA-100을 만들 수 있다. 이는 그림 5에 제시되어 있다.

제안된 알고리즘은 곱셈을 0회 수행하므로, 곱셈을 수행하는 Karatsuba와 Toom-3 곱셈법과 알고리즘 복잡도를 단순 비교 하지는 못한다. 다만, 제안된 알고리즘은 $n+1$ 자리수에 대한 덧셈을 $l(b)+8$회 수행한다. 즉, $l(b)=n$이면 $(n+1)(n+7)=n+8n+7$로 (n^2)의 덧셈만을 수행한다.

Ⅴ. 결론

본 논문은 $a\times b=c$의 곱셈을 수행함에 있어 완전 덧셈만으로 곱셈을 효율적으로 수행하는 자리이동-덧셈법을 제안하였다.

제안된 방법은 Karatsuba와 Toom-3와 같이 일반적인 곱셈 횟수를 1회 줄이는데 초점을 맞추지 않고, 곱셈을 전혀 수행하지 않는 방법에 초점을 두었다.

제안된 자리이동-덧셈법은 초등수학 곱셈법의 $n\times n=n^2$ 자리수의 곱셈과 $(n+1)n=n^2+n$ 자리수 덧셈을 $(n+1)(n+7)=n^2+8n+7$의 덧셈만을 수행하는 방법으로 성능을 향상시켰다.

References

[1] Wikipedia, "List of Unsolved Problems in Computer Science," http://en.wikipedia.org/wiki/List_of_unsolved_problems_in_computer_science, Wikimedia Foundation Inc., 2012.

[2] Wikipedia, "Multiplication Algorithm," http://en.wikipedia.org/wiki/Multiplication_algorithm, Wikimedia Foundation Inc., 2012.

[3] A. Karatsuba and Y. Ofman, "Multiplication of Multidigit Numbers on Automata," Soviet Physics-Doklady, Vol. 7, pp. 595-596, 1963.

[4] A. Mandal and R. Syal, "Tripartite Modular Multiplication using Toom-Kook Multiplication," International Journal of Advanced Research in Computer Science and Electronics Engineering, Vol. 1, No. 2, pp. 100-104, 2012.

[5] R. P. Brent and P. Zimmermann, "Modern Computer Arithmetic, Version 0.5," Cambridge Monographs on Computational and Applied Mathematics, Cambridge University Press, 2010.

[6] A. Eigenwilling and K. Mehlhorn, "Multiplication of Long Integers (Faster than Long Multiplication)," Max Planck Institute for Informatics, Saarbrücken, Germany, http://www.mpi-inf.mpg.de/~mehlhorn/ftp/ chapter 2A-en.pdf, 2005.

소개

상 운(정회원)

- 1987년: 한국항공대학교 항공전자공학과 (학사)
- 1997년: 경상대학교 컴퓨터과학과 (석사)
- 2001년 : 경상대학교 컴퓨터과학과 (박사)
- 2003년 : 강원도립대학 컴퓨터응용과 전임강사
- 2004년 ~ 2007.2 : 국립 원주대학 여성교양과 조교수
- 2007.3 ~ 현재 : 강릉원주대학교 멀티미디어공학과 부교수
- 관심분야 :소프트웨어 프로젝트 관리, 개발 방법론, 분석과 설계 방법론, 시험 및 품질보증, 소프트웨어 신뢰성, 그래프 알고리즘
- e-mail : sulee@gwnu.ac.kr