RAIRO. INFORMATIQUE THÉORIQUE ET APPLICATIONS

SREĆKO BRLEK
PIERRE CASTÉRAN
LAURENT HABSIEGER
RICHARD MALLETTE

On-line evaluation of powers using Euclid's algorithm

RAIRO. Informatique théorique et applications, tome 29, n° 5 (1995), p. 431-450

<http://www.numdam.org/item?id=ITA_1995__29_5_431_0>

Informatique théorique et Applications/Theoretical Informatics and Applications
(vol. 29, n° 5, 1995, p. 431 à 450)

ON-LINE EVALUATION OF POWERS USING EUCLID'S ALGORITHM (*)

by Srećko BRLEK ([1]), Pierre CASTÉRAN ([2]),
Laurent HABSIEGER ([3]) and Richard MALLETTE ([1])

Communicated by R. CORI

Abstract. – *The aim of this paper is to present an efficient algorithm to compute powers of an element in a semigroup. The problem to compute x^n using a minimal number of semigroup operations is equivalent to the problem of computing an optimal addition chain for n. The algorithm presented here is based upon a suitable continued fraction expansion for n and is remarkably stable: the asymptotic length of the addition chain produced is bounded by $1,5 \log_2(n)$ for the worst case, and the average length is asymptotically $\approx 1,2946 \log_2(n)$.*

Résumé. – *Cet article présente un algorithme efficace de calcul des puissances d'un élément x donné dans un semi-groupe. Le problème du calcul de x^n en un nombre minimal de multiplications est équivalent au calcul d'une chaîne d'addition optimale pour n. L'algorithme que nous étudions, basé sur un développement approprié de n en fractions continues est remarquablement stable : la longueur asymptotique d'une chaîne ainsi produite est bornée par $1.5 \log_2(n)$ pour le pire cas; la longueur moyenne est asymptotiquement $\approx 1.2946 \log_2(n)$.*

1. INTRODUCTION

Following Knuth [10], we define an *addition chain* for a positive integer n to be a sequence $C = (n_0, n_1, \ldots, n_s)$ of positive integers such that

(i) $n_0 = 1$ and $n_s = n$,

(ii) for each i, $1 \leq i \leq s$, there exist k, $j < i$ such that $n_i = n_j + n_k$.

The integer s is the *length* of the chain C and is denoted by $|C|$. The *chain length* $\ell(n)$ *of* n is the minimal length of all possible chains for n.

(*) Received revised version July 1994.

([1]) LACIM, Département de Maths et Info, Université du Québec à Montréal, C.P. 8888, Succ. "Centre Ville", Montréal, Québec, H3C 3P8 Canada.

([2]) LABRI, Université Bordeaux 1, 351, Cours de la Libération, 33405 Talence Cedex, France.

([3]) Laboratoire d'algorithmique arithmétique, CNRS, UM9936, Université Bordeaux 1, 351, Cours de la Libération, 33405 Talence Cedex, France.

Informatique théorique et Applications/Theoretical Informatics and Applications
0988-3754/95/05/$ 4.00/

We also denote the chain length of a set of numbers $\{m_1, \ldots, m_t\}$ by $\ell(m_1, \ldots, m_t)$. Addition chains for n generate multiplication schemes for the computation of x^n. For instance, the chain (1, 2, 4, 8, 9, 17, 34, 43) leads to the following scheme for the computation of x^{43}:

$$xx = x^2, \quad x^2 x^2 = x^4, \quad x^4 x^4 = x^8, \quad xx^8 = x^9,$$
$$x^8 x^9 = x^{17}, \quad x^{17} x^{17} = x^{34}, \quad x^9 x^{34} = x^{43}.$$

Therefore, $\ell(n)$ is equal to the smallest number of multiplications required for the computation of x^n. Any explicit algorithm for the generation of addition chains clearly sets an upper bound on the function $\ell(n)$. Thus the usual binary expansion algorithm (*see* [10]) implies that $\ell(n) \leq \lambda(n) + \nu(n) - 1$, where $\lambda(n) = \lfloor \log_2(n) \rfloor$ and $\nu(n)$ is the number of 1 in the binary expansion of n. However, the problem of computing the exact value of $\ell(n)$ seems to be difficult. Indeed, a slightly more complex problem, namely the problem of computing the chain length for a set of integers, has been shown to be NP-complete [9]. Therefore, it is interesting to consider sub-optimal addition chains, provided that they can be constructed in an efficient way. We briefly recall the major results about the length of addition chains. Schönhage [12] established the lower bound $\ell(n) \geq \log_2(n) + \log_2(\nu(n)) - 2.13$. Brauer [1] showed that the optimal length was asymptotically $\lambda(n) = \lfloor \log_2(n) \rfloor$ by producing the upper bound, $\ell(n) \leq \lambda(n) + \lambda(n)/p + 2^p - 2$ for all $p \geq 1$. For a suitable p (for instance, $p = \lfloor 1/2 \log_2(\lambda(n)) \rfloor$, the right-hand side of the inequality converges to $\lambda(n)$. Thurber [13] improved Brauer's result as follows: take first the binary representation n; set $m = \lambda(n) + 1$; then, starting from the leading digit, partition the binary word into equal parts of length p, producing the set $\{n_1, n_2, \ldots, n_{\lceil m/p \rceil}\}$. Then, each n_i is in the initial set $\{1, 2, 3, \ldots, 2^p - 1\}$, and the number n is produced by applying the rules:

R1) $M_1 = n_1$;

R2) $M_i = M_{i-1} 2^p + n_i$, for $i = 2, \ldots, \lceil m/p \rceil - 1$;

R3) $M_{\lceil m/p \rceil} = n = M_{\lceil m/p \rceil - 1} 2^{m - p(\lceil m/p \rceil - 1)} + n_{\lceil m/p \rceil}$.

Note that multiplication by 2^p is achieved by shifting, and if n_i is even, then it can be replaced by an odd number n_i' such that $n_i = n_i' 2^j$ for some $j \geq 1$. This does not affect the total number of shifts. Then $\ell(n)$ is bounded by the total number of operations needed to produce n, namely

$$\ell(n) \leq \lambda(n) - (p-1) + \lfloor \lambda(n)/p \rfloor + 2^{(p-1)},$$

where $2^{(p-1)}$ stands for the computation of all odd numbers less than 2^p. In that same paper, Thurber points out that this construction can be improved for small values of n: he proposed to take greater values of p and to replace the initial set of $2^{(p-1)}$ odd numbers by a chain for the set $\{n_1, n_2, \ldots, n_{\lceil m/p \rceil}\}$, yielding the upper bound

$$\ell(n) \leq \lambda(n) - (p-1) + \lfloor \lambda(n)/p \rfloor + \ell(n_1, n_2, \ldots, n_{\lceil m/p \rceil}).$$

Thus, the problem of computing $\ell(n)$ is reduced to the computation of an optimal chain for the set $\{n_1, n_2, \ldots, n_{\lceil m/p \rceil}\}$. This suggests the use of Yao's [14] method. Yao's algorithm is asymptotically optimal, but there is still place for improvements when the numbers n_i are small.

In [2, 4], we have introduced such an algorithm for the case of a set of two numbers. Namely, a chain for $\{n, k\}$ is obtained through the continued fraction expansion for n/k, where $n > k$. The general case is treated in [5]. We called chains of this form *continued fraction addition chains, cf-chains* for short, or also *euclidean* chains. We proved that for an infinite class of integers, cf-chains are much closer to optimal addition chains than the chains obtained by the usual binary method. Minimal-length euclidean chains are not optimal but they thave the nice property of being easy to compute and are significantly shorter, on the average, than chains obtained by the binary method (section 3). Indeed, the worst case for the binary method occurs when $n = 2^k - 1$, and yields a chain of length $2\lambda(n)$ while the average case length is $1.5\lambda(n)$. Asymptotically, our algorithm produces addition chains with length bounded by $1.2946 \log_2(n)$ for the average case, $1.5 \log_2(n)$ for the worst case, and is asymptotically optimal for the worst case of the binary method.

2. THE CONTINUED FRACTION ALGORITHM

Let $\mathcal{C}(n) = (n_0, n_1, \ldots, n_s)$ and $\mathcal{C}(m) = (m_0, m_1, \ldots, m_t)$ be some addition chains for n and m respectively. Let j be also one of the integers appearing in $\mathcal{C}(n)$. Define the product $\mathcal{C}(n) \otimes \mathcal{C}(m)$ and the sum $\mathcal{C}(n) \oplus j$ to be

$$\mathcal{C}(n) \otimes \mathcal{C}(m) = (n_0, n_1, \ldots, n_s, nm_1, nm_2, \ldots, nm_t),$$
$$\mathcal{C}(n) \oplus j = (n_0, n_1, \ldots, n_s, n_s + j).$$

The following lemma is immediate.

LEMMA 1: *Let $n = aq + r$ with $r < a$. If $C(a, r)$ is an addition chain for $\{a, r\}$ and if $C(q)$ is an addition chain for q, then, $C(a, r) \otimes C(q) \oplus r$ is an addition chain for n.* ■

In any case, the chain for 2^a should always be $(1, 2, 4, \ldots, 2^a)$, since it is clearly the unique minimal length chain.

2.1. The algorithm

The basic idea of the algorithm is to split the binary word representing n as follows. Let $p = \lfloor(\lambda(n) + 1)/2\rfloor$, $k = \lfloor n/2^p\rfloor$ and $r = n \bmod 2^p$. Then $n = k \cdot 2^p + r$. If the number $m = \lambda(n) + 1$ of digits of n is odd, then we have $k > r$ and by Lemma 1, we obtain the chain

$$C(n) = C(k, r) \otimes C(2^p) \oplus r.$$

If m is odd it may happen that $k \leq r$, in which case one rewrites $n = k \cdot (2^p + 1) + (r - k)$. Clearly, $r - k < k$ and we apply again Lemma 1 to produce the chain

$$C(n) = C(k, r - k) \otimes C(2^p + 1) \oplus (r - k).$$

Observe that $p = \lfloor\#\text{digits}/2\rfloor$. For this reason we call our method *dichotomic*. We give now a precise form to the method as an algorithm.

Algorithm Chain (n)
  if $n = 2^a$ *then return* $(1, 2, 4, \ldots, 2^a)$
  elif $n = 3$ *then return* $(1, 2, 3)$
  else let $p = \lfloor(\lambda(n) + 1)/2\rfloor$; $k = n$ *div* 2^p; $r = n \bmod 2^p$
    if $r \geq k$ *then return* *Euclid* $(k, r - k) \otimes$ *Chain* $(2^p + 1) \oplus (r - k)$
    else return *Euclid* $(k, r) \otimes$ *Chain* $(2^p) \oplus r$
    endif
  endif
end

The length of chains obtained by the algorithm *Chain* will be denoted by $\ell(n, \sigma)$.

Algorithm Euclid (n_1, n_2)
  let $q = n_1$ *div* n_2; $r = n_1 \bmod n_2$
  if $r = 0$ *then return* *Chain* $(n_2) \otimes$ *Chain* (q)
  else return *Euclid* $(n_2, r) \otimes$ *Chain* $(q) \oplus r$
  endif
end.

The length of chains obtained by algorithm *Euclid* will be denoted by $L(n_1, n_2)$. Observe that in algorithm *Chain*, the test "if $r \geq k$" can be avoided if one replace the *else* part by

else let $p = \lfloor(\lambda(n)+1)/2\rfloor$; $k = \lfloor n/2^p \rfloor$;
return $Euclid(n, k)$

LEMMA 2: *For all* $n \in N^*$, *let* $k = \left\lfloor n/2^{\lfloor(\lambda(n)+1)/2\rfloor} \right\rfloor$. *Then*

$$\ell(n, \sigma) = L(n, k).$$

Proof: Let $\sigma(n) = k$. We only need to check this for $n = 3$ and $n = 2^a$ $(a = 0, 1, \ldots)$. Since $\sigma(3) = 1$ we have $L(3, 1) = \ell(3) + \ell(1) = \ell(3)$ and the relation is true for $n = 3$. For powers of 2, we distinguish between even and odd powers. We have $\sigma(2^{2a}) = \sigma(2^{2a-1}) = 2^a$ from which we obtain

$$L(2^{2a}, 2^a) = 2\,\ell(2^a) = 2a$$

and

$$L(2^{2a-1}, 2^a) = \ell(2^a) + \ell(2^{a-1}) = 2a - 1. \quad \blacksquare$$

As a consequence, the length of chains produced by the dichotomic algorithm is

$$\ell(n, \sigma) = L(n, k) = \ell(d, \sigma) + s - 1 + \sum_{i=1}^{s} \ell(q_i, \sigma). \tag{1}$$

where $d = Gcd(n, k)$ and s is the length of the continued fraction $n/k = [q_1, \ldots, q_s]$.

Example 1: For $n = 5514$, we have $p = 6$, $k = 86$ and $r = 10$. Hence the euclidean chain produced by the algorithm described above is

$(1, 2) \otimes (1, 2) = (1, 2, 4)$,
$(1, 2, 4) \oplus 2 = (1, 2, 4, 6)$,
$(1, 2, 4, 6) \oplus 4 = (1, 2, 4, 6, 10)$,
$(1, 2, 4, 6, 10) \otimes (1, 2, 4, 8) = (1, 2, 4, 6, 10, 20, 40, 80)$,

$(1, 2, \ldots, 80) \oplus 6 = (1, 2, 4, 6, 10, 20, 40, 80, 86),$
$(1, 2, \ldots, 80, 86) \otimes (1, 2, 4, 8, 16, 32, 64)$
$= (1, 2, \ldots, 80, 86, 172, 344, 688, 1376, 2752, 5504),$
$(1, 2, \ldots, 5504) \oplus 10$
$= (1, 2, 4, 6, 10, 20, 40, 80, 86, 172, 344, 688, 1376, 2752, 5504, 5514).$

Observe that this chain has length 15, whereas the binary method produces the chain

$(1, 2, 4, 5, 10, 20, 21, 42, 43, 86, 172, 344, 688, 689, 1378, 2756, 2757, 5514)$

which has lenght 17.

Example 2: For $n = 3067$, we have $p = 6$, $k = 47$ and $r = 59$, and the chain produced here is

$(1, 2) \otimes (1, 2, 4, 5) = (1, 2, 4, 8, 10),$
$(1, 2, 4, 8, 10) \oplus 1 = (1, 2, 4, 8, 10, 11),$
$(1, 2, 4, 8, 10, 11) \oplus 1 = (1, 2, 4, 8, 10, 11, 12),$
$(1, 2, 4, 8, 10, 11, 12) \otimes (1, 2, 3) = (1, 2, 4, 8, 10, 11, 12, 24, 36),$
$(1, 2, 4, 8, 10, 11, 12, 24, 36) \oplus 11 = (1, 2, 4, 8, 10, 11, 12, 24, 36, 47),$
$(1, 2, 4, 8, 10, 11, 12, 24, 36, 47) \otimes (1, 2, 4, 8, 16, 32, 64, 65)$
$= (1, 2, 4, 8, 10, 11, 12, 24, 36, 47, 94, 188, 376,$
$752, 1504, 3008, 3055),$
$(1, 2, 4, 8, 10, 11, 12, 24, 36, 47, 94,$
$188, 376, 752, 1504, 3008, 3055) \oplus 12$
$= (1, 2, 4, 8, 10, 11, 12, 24, 36, 47, 94,$
$188, 376, 752, 1504, 3008, 3055, 3067).$

This chain has length 17, whereas the binary method produces the chain with length 20,

$(1, 2, 4, 5, 10, 11, 22, 23, 46, 47, 94, 95, 190, 191, 382, 383,$
$766, 1532, 1533, 3066, 3067).$

3. RESULTS

3.1. Comparison with optimal chains for small numbers

Table 1 shows that on the average, the binary method is by far the *worst* with respect to the length. The computation time was obtained by running the program on a MIPS2000 and represents the time needed to produce all the addition chains. Note that in order to compute x^n we don't really need the actual chain but only the associated multiplication scheme which is better described in computer arithmetics by a *dag* (directed acyclic graph). This will be discussed in section 3.3.

TABLE 1
Running time comparison of algorithms.

Strategy	Total length ($N = 1000$)	Difference with ℓ	Time (min./sec.)
Binary	11925	1117	$\ll 0.1$ s
Dichotomic	11064	256	$\ll 0.3$ s
Optimal	10808	0	340 min.

For a given method γ (binary, dichotomic, optimal), let $\ell(n, \gamma)$ be the chain length for n obtained by γ. Let k be fixed. We shall denote the average length and maximum values of $\ell(n, \gamma)$, for $n \in [2^k, 2^{k+1} - 1]$, by $\overline{\ell_k(n, \gamma)}$ and $M(k, \gamma)$, respectively.

For the average and worst cases, the dichotomic method is by far better than the binary method as can be seen in Table 2. We have not included the values $n \leq 511$ since the maximum value is then $1.75\,\lambda(n)$ obtained for $n = 11$ for all methods. Clearly, for very small values of n, the comparison is not relevant.

TABLE 2
Average and maximum values for $512 \leq n \leq 1023$.

Strategy	$\overline{\ell_k(n, \gamma)}/\lambda(n)$	$M(k, \gamma)/\lambda(n)$
Binary	1.5	2.0
Dichotomic	1.3812	1.5556
Optimal	1.3483	1.4444

We define Thurber's upper bound $T(k)$ by

$$T(k) = \min_{p \in [1, k]} \{k - (p-1) + \lfloor k/p \rfloor + 2^{p-1}\}.$$

Table 3 compares the statistics $\overline{\ell_k(n, \sigma)}/\lambda(n)$, $M(k, \sigma)$, $T(k)$.

TABLE 3
Chain length statistics for values of n in $[2^k, 2^{k+1}-1]$.

k	12	13	14	15	16	17	18	19	20
$\overline{\ell_k(n, \sigma)}/k$	1.3586	1.3568	1.3490	1.3486	1.3431	1.3426	1.3380	1.3377	1.3338
$M(k, \sigma)$	18	20	21	23	24	26	27	29	30
$T(k)$	18	19	20	22	23	24	26	27	28
$M(k, \sigma)/k$	1.5	1.5384	1.5	1.5333	1.5	1.5294	1.5	1.5263	1.5

As expected, computer calcultations show that the average value $\overline{\ell_k(n, \sigma)}/k$ for the dichotomic method decreases with increasing n, while the maximum remains fairly stable near $1.5\,\lambda(n)$. Note that Thurber's method requires an analysis of the binary representation of n, in order to produce the necessary shifts. Moreover, the unnecessary odd numbers can be deleted at a cost $\mathbf{O}(\lambda(n))$. To make the comparison complete, it would be interesting to produce an average case analysis of Thurber's method. As far as we know this analysis is not known and goes beyong the scope of this paper.

Now we shall produce an upper bound for $\ell(n, \sigma)$. To do so we need a preliminary result.

PROPOSITION 1: *Let a be a positive integer and assume that*

$$L(n, k) \leq 2\log_2 n - 1 \quad \textit{for} \quad 2 \leq n \leq 2^{a+1} - 1 \quad \textit{and} \quad 1 \leq k < n.$$

Then, $\ell(q, \sigma) \leq 2\log_2(q+1) - 2$, *for* $1 \leq q \leq 2^{2a} - 1$.

Proof: It is straightforward to check this inequality for $1 \leq q \leq 63$. For a given q such that $64 \leq q \leq 2^{2a} - 1$, let $k = \sigma(q)$ and $b = \lambda(k)$. Then,

$$q \in [2^{b-1}k,\ 2^{b-1}k + 2^{b-1} - 1] \cup [2^b k,\ 2^b k + 2^b - 1]$$

where $k \in [2^b, 2^{b+1} - 1]$. It follows that $2^6 \leq q < 2^{2b+1}$ from which we obtain $b > 5/2$ and, thus, $b \geq 3$. In the same way $2^{2b-1} \leq q < 2^{2a}$ yields $b \leq a$. An easy consequence is that $8 \leq k \leq 2^{a+1} - 1$ and, by the assumption

$$L(k, r) \leq 2\log_2 k - 1, \tag{2}$$

for $1 \leq r \leq k - 1$. As a special case we have

$$\ell(k, \sigma) = L(k, 1) \leq 2\log_2 k - 1.$$

Case 1: $2^{b-1}k = q$. Then $\ell(q, \sigma) = L(q, k) = \ell(2^{b-1}, \sigma) + \ell(k, \sigma)$, and

$$\ell(q, \sigma) \leq b - 1 + 2\log_2 k - 1 \leq 2b - 3 + 2\log_2 k - 1 < 2\log_2(q+1) - 2.$$

Case 2: $2^{b-1}k < q < 2^{b-1}k + 2^{b-1}$. Then, $q = 2^{b-1}k + r$ where $0 < r < 2^{b-1} < k$. Therefore, using (2),

$$\ell(q, \sigma) = L(q, k) = \ell(2^{b-1}, \sigma) + 1 + L(k, r) \leq b + 2\log_2 k - 1.$$

Now, it is easy to check that $k < (2^{b+1}k)^{1/2} < (4q)^{1/2}$ and $2^{b-1}k < q$. This yields the inequality

$$\ell(q, \sigma) \leq 3/2\log_2 q + 1.$$

But $3/2\log_2 q + 1 \leq 2\log_2 q - 2$ for $q \geq 2^6$ and, thus,

$$\ell(q, \sigma) \leq 2\log_2(q+1) - 2.$$

Case 3: $q = 2^b k$. Then, $\ell(q, \sigma) = L(q, k) = b + \ell(k, \sigma)$, and

$$\ell(q,\sigma) \leq b-1+2\log_2 k \leq 2b-2+2\log_2 k = 2\log_2 q-2 < 2\log_2(q+1)-2.$$

Case 4: $2^b\sigma < q < 2^b k + 2^b$. Then we have $q = 2^b k + r$ where $0 < r < 2^b \leq k$. Using (2)

$$\ell(q, \sigma) = L(q, k) = \ell(2^b, \sigma) + 1 + L(k, r) \leq b + 2\log_2 k.$$

But we also have $k < (2^{b+1}k)^{1/2} < (2q)^{1/2}$ and $2^b k < q$, yielding the inequality

$$\ell(q, \sigma) < 3/2\log_2 q + 1/2\log_2 q - 2 < 2\log_2(q+1) - 2,$$

using the fact that $q \geq 2^5$. ■

THEOREM 1: *Let* $n \geq 2$. *Then,* $L(n, k) \leq 2\log_2 n - 1$, *for* $1 \leq k < n$.

Proof: We proceed by induction on n. The property can be checked for $n = 2, 3, 4, 5, 6, 7$. Now, observe that for $k = 1$, the claim is true since

$$L(n, 1) = \ell(n, \sigma) \leq 2\log_2(n+1) - 2 \leq 2\log_2 n - 1,$$

for $n \geq 8$. Assuming the chaim true for $2 \leq n \leq 2^{a+1} - 1$ we show that it remains true for $2^{a+1} \leq n \leq 2^{a+2} - 1$, where $a > 2$. Then we have $2a \geq a + 2$ and we can apply Proposition 1.

Let n and k be integers such that $2^{a+1} \leq n \leq 2^{a+2}-1$, and $1 < k < n$. Let $d = Gcd(n, k)$ and $[q_1, \ldots, q_l]$ be the continued fraction expansion of n/k.

Case 1: $Gcd(n, k) > 1$. If $l = 1$ then $n = q_1 k$ and

$$L(n, k) = \ell(q_1, \sigma) + \ell(k, \sigma) = L(q_1, 1) + L(k, 1),$$

where $\max(q_1, k) \leq n/\min(q_1, k) \leq n/2 < 2^{a+1}$. Using the induction hypothesis we obtain

$$L(n, k) \leq 2\log_2 q_1 - 1 + 2\log_2 k - 1 < 2\log_2 n - 1.$$

If $l \neq 1$ then from (1) we have $L(n, k) = \ell(d, \sigma) + L(n', k') + 1$ and again, by the induction hypothesis

$$L(n, k) \leq 2\log_2 d - 1 + 2\log_2 n' - 1 + 1 = 2\log_2 n - 1.$$

Case 2: $Gcd(n, k) = 1$ and $l = 2$. Then $n = (q_1 q_2 + 1)$ and $k = q_2$. Moreover, $L(n, k) = \ell(q_1, \sigma) + \ell(q_2, \sigma) + 1$. Using Proposition 1 we have the upper bound

$$\begin{aligned} L(n, k) &\leq 2\log_2 (q_1 + 1)(q_2 + 1) - 3 \\ &= 2\log_2 n - 1 + 2\log_2 \left(1 + \frac{q_1 + q_2}{q_1 q_2 + 1}\right) - 2. \end{aligned}$$

But $q_1 + q_2 \leq 1 + q_1 q_2$, therefore

$$\left(1 + \frac{q_1 + q_2}{q_1 q_2 + 1}\right) \leq 2.$$

and thus $L(n, k) \leq 2\log_2 n - 1$.

Case 3: $Gcd(n, k) = 1$ and $\ell > 2$. We have $n = (q_1 q_2 + 1) n' + q_1 k'$ and $k = (q_2 + 1) n' + k'$, for some integers n', k' such that $1 \leq k' < n'$. Moreover $n' < n/(q_1 q_2 + 1) \leq n/2 < 2^{a+1}$, and we can apply the induction hypothesis to n'. Since $(q_1 + 1)(q_2 + 1) \leq 2(q_1 q_2 + 1)$ it follows that

$$\begin{aligned} L(n, k) &= \ell(q_1, \sigma) + \ell(q_2, \sigma) + 2 + L(n', k') \\ &\leq 2\log_2 (q_1 + 1)(q_2 + 1) n' - 3 \\ &\leq 2\log_2 (2n) - 3 = 2\log_2 n - 1. \end{aligned}$$

This completes the proof. ■

COROLLARY 1: *For every positive n, let $m = \lambda(n) + 1$. Then*

$$\ell(n, \sigma) \leq m + \left\lfloor \frac{m}{2} \right\rfloor.$$

Proof: Let $p = \lfloor m/2 \rfloor$ and $a = \lfloor n/2^p \rfloor$. Then $n = aq + r$, with $r < a$. Using Lemma 2 we have,

$$\ell(n, \sigma) \leq L(a, r) + \ell(q, \sigma) + 1 \leq \lfloor 2 \log_2 a \rfloor + \ell(q, \sigma),$$

where $\lfloor 2 \log_2 (a) \rfloor$ is used instead of $2 \log_2 a$. This is justified since we only deal with integers. Since $n \geq a \cdot 2^p$ the following inequalities hold.

$$\lfloor 2 \log_2 a \rfloor \leq \lfloor 2 \log_2 (n) \rfloor - 2 \lfloor m/2 \rfloor \leq 2 \lambda(n) + 1 - 2 \lfloor m/2 \rfloor.$$

Therefore,

$$\ell(n, \sigma) \leq 2(m-1) + 1 - 2 \lfloor m/2 \rfloor + \ell(q, \sigma).$$

If m odd, then $q = 2^{\lfloor m/2 \rfloor}$ and $\ell(q, \sigma) = \lfloor m/2 \rfloor$. Hence,

$$\ell(n, \sigma) \leq 2(m-1) + 1 - 2\,\frac{m-1}{2} + \lfloor m/2 \rfloor = m + \left\lfloor \frac{m}{2} \right\rfloor.$$

If m even, then $q = 2^{\lfloor m/2 \rfloor}$ or $q = 2^{\lfloor m/2 \rfloor} + 1$. It is straightforward to check the inequality $\ell(q, \sigma) \leq \lfloor m/2 \rfloor + 1$. Hence,

$$\ell(n, \sigma) \leq 2(m-1) + 1 - 2\,\frac{m}{2} + \lfloor m/2 \rfloor + 1 = m + \left\lfloor \frac{m}{2} \right\rfloor. \quad \blacksquare$$

COROLLARY 2: *For every positive* n, $\ell(n, \sigma) \leq 1.5 \log_2 n + 1$.

We conclude this section by showing that this upper bound is thight. First, we have $\ell(39, \sigma) = 8 = 1.513602876 \log_2 (39)$.

Now, any couple (a, b) of integers such that $a > b > 0$ and satisfying $L(a, b) = \lfloor 2 \log_2 a \rfloor - 1$, allows the construction of a near worst-case number. Indeed, set $n = a \cdot 2^{\lambda(a)} + b$, then

$$\begin{aligned} \ell(n, \sigma) &= L(a, b) + \lambda(a) + 1, \\ &= \lfloor 2 \log_2 (a) \rfloor - 1 + \lambda(a) + 1, \\ &\geq 3 \cdot \lambda(a). \end{aligned}$$

Set $m = \lambda(n) + 1$ and observe that $m = 2 \cdot (\lambda(a) + 1)$. Therefore

$$\ell(n, \sigma) \geq m + \left\lfloor \frac{m}{2} \right\rfloor - 1.$$

It remains to prove that arbitrarily large couples (a, b), for which $L(a, b)$ is close to this bound, exist. A first example is provided by the Fibonacci sequence. The continued fraction corresponding to two consecutive Fibonacci numbers is

$$(F_n, F_{n-1}) = [1, 1, 1, \ldots, 1].$$

Then, according to (1), $L(F_n, F_{n-1}) = s - 1 = 1.44035 \log_2 (F_n) + 0.672$, where s is the length of the continued fraction (*see* [10]).

In order to get near worst-case numbers we need a set of good candidates. Namely numbers such that, $\ell(q, \sigma)$ is close to the upper bound given in Theorem 1. A straightforward verification shows the next Lemma.

LEMMA 3: *The following conditions hold for* $\ell(q, \sigma)$.

(i) $\ell(q, \sigma) \geq 2\log_2(q) - 2$, *for* $q \in \{1, 2, 3, 4, 5, 7, 11\}$;

(ii) $\ell(q, \sigma) \geq 2\log_2(q) - 3$,

for $q \in [1, 11] \cup \{13, 14, 15, 19, 21, 22, 29, 30, 31, 39\}$.

This result leads to the following generalization.

PROPOSITION 2: *Let* $a > b$ *be such that* $L(a, b) \geq 2\log_2 a - c$, *for some* $c > 1$ *and let* $n = aq + b$ *for some* $q \in \{1, 2, 3, 4, 5, 7, 11\}$. *Then*

$$L(n, a) \geq 2\log_2 n - (c + 2), \qquad \text{if} \quad q \geq \frac{b}{a} \cdot \frac{1}{\sqrt{2} - 1}.$$

Proof: By definition,

$$\begin{aligned} L(n, a) = L(a, b) + \ell(q, \sigma) + 1) &\geq 2\log_2 a - c + 2\log_2 q - 2 + 1 \\ &= 2\log_2 (aq) - (c + 1). \end{aligned}$$

In order to get $2\log_2 (aq) - (c + 1) \geq 2\log_2 (aq + b) - (c + 2)$, we need to solve the following inequality

$$2\log_2 (aq + b) - 2\log_2 (aq) \leq 1.$$

A simple verification shows that this last inequality is satisfied when $q \geq \frac{b}{a} \cdot \frac{1}{\sqrt{2} - 1}$. ■

A direct adaptation of the proof above yields the following result.

COROLLARY 1: *Let* $a > b$ *be such that* $L(a, b) \geq 2\log_2 a - c$, *for some* $c > 1$, *and let* $n = a + b$. *Then*

$$L(n, a) \geq \begin{cases} 2\log_2 n - c, & \text{if} \quad b < a(\sqrt{2} - 1), \\ 2\log_2 n - (c + 1), & \text{otherwise} \end{cases}$$

Observe that there is still a gap between the numbers obtained and the experimental results listed in Table 3. A set of good candidates for the worst case is given by continued fractions of the following type:

$$[1, 11, 1, 7, \ldots, 1, 7]; \quad (L(915, 844)/\log_2(915) = 1.829709371);$$
$$[1, 7, 1, 11, 1, 7, \ldots, 1, 7]; \quad (L(919, 816)/\log_2(919) = 1.828539655);$$
$$[1, 7, 1, 7, \ldots, 1, 7]; \quad (L(631, 560)/\log_2(631) = 1.827662954).$$

It seems that the worst case for $n \in [2^k, 2^{k+1} - 1]$ is given by a continued fraction of the first type. We will see in the next section that this agress with the asymptotic analysis.

3.2. Asymptotic values

Let $[q_1, \ldots, q_l]$ be some continued fraction expansion. The associated matrix A is

$$A = \begin{pmatrix} q_1 & 1 \\ 1 & 0 \end{pmatrix} \cdots \begin{pmatrix} q_l & 1 \\ 1 & 0 \end{pmatrix}.$$

Let $t = \mathrm{Tr}(A)$ be the trace of A. Since $\det A = (-1)^l$, the caracteristic polynomial of A is $X^2 - tX + (-1)^l$. The absolute value of the product of the eigenvalues of A is 1. The only case where $\mathrm{Tr}(A) \leq 2$ occurs when $l = 1$ and $q \in \{1, 2\}$. This implies that in any case, the matrix A has an eigenvalue strictly greater than 1 and an eigenvalue whose absolute value is strictly less than 1. The greatest of the eigenvalues is denoted r. Its value is

$$r = \frac{t + \sqrt{t^2 - 4(-1)^l}}{2}.$$

Define a sequence n_m, k_m by

$$\begin{pmatrix} n_m \\ k_m \end{pmatrix} = A^m \begin{pmatrix} 1 \\ 0 \end{pmatrix},$$

so that the continued fraction expansion of (n_m, k_m) is $[q_1, \ldots, q_l]$ repeated m times, and such that $Gcd(n_m, k_m) = 1$. Then, for $m \geq 1$

$$L(n_m, k_m) = \begin{cases} m(\ell(q_1, \sigma) + \ldots + \ell(q_l, \sigma)) + ml - 1, & \text{if } q_l > 1, \\ m(\ell(q_1, \sigma) + \ldots + \ell(q_l, \sigma)) \\ \quad + ml - 1 + \ell(q_{l-1} + 1, \sigma) - 1, & \text{if } q_l = 1. \end{cases}$$

We have $L(n_m, k_m) \sim m(l(q_1, \sigma) + \ldots + \ell(q_l, \sigma) + l)$. But $n_m \sim ar^m$, for some a, and

$$L(n_m, k_m) \sim C \log_2(n_m) \quad \text{where} \quad C = \frac{\ell(q_1, \sigma) + \ldots + \ell(q_l, \sigma) + l}{\log_2 r}.$$

Therefore, to optimize the constant C we need to choose q_i such that the length $\ell(q_i, \sigma)$ is realized for the first time, namely for the values $q_i \in \{1, 2, 3, 5, 7, 11, 19, 29, \ldots,\}$. These numbers are precisely given by the function $c(r)$ in Knuth [10].

Example 3: The asymptotic bound for periodic expansions can be easily computed.

1) Let $q_1 = q$. Then $r = \dfrac{q + \sqrt{q^2 + 4}}{2}$. (ref: the Fibonacci sequence seen earlier)

2) $l = 2$. We have $r = \dfrac{q_1 q_2 + 2 + \sqrt{q_1 q_2 (q_1 q_2 + 4)}}{2}$. The study of worst cases for small numbers shows the importance of the periodicity (1, 7). For $q_1 = 1$ and $q_2 = 7$, $C = 1.9036\ldots$

Extensive computations motivate the following conjectures, which are related,

CONJECTURES

$$\overline{\lim_{a \to \infty}} \max_{1<b<a} \frac{L(a, b)}{\log_2 a} = 1.9036\ldots; \quad \overline{\lim_{n \to \infty}} \frac{\ell(n, \sigma)}{\log_2 n} = 1.4518\ldots$$

Nevertheless, from Corollary 2 of Theorem 1 we have the following asymptotic upper bound,

$$\overline{\lim_{n \to \infty}} \frac{\ell(n, \sigma)}{\log_2 n} \leq 1.5.$$

In [3] we established the equality $\ell(2^n - 1, \sigma) = n - 2 + \lambda(n) + \nu(n)$. It shows that, for the worst case of the binary method, the chains produced by the dichotomic method are asymptotically optimal. Namely,

$$\lim_{n \to \infty} \frac{\ell(2^n - 1, \sigma)}{\lambda(2^n - 1)} = 1.$$

Given a fixed k, let $P(r)$ be the probability that a quotient of r occurs in the computation of the continued fraction of k/a when a is chosen at random.

The average length produced by the dichotomic method is asymptotically $\approx 1.2946 \log_2 (n)$. The proof of this fact is established following the analysis of the average length of a continuous fraction as described in Knuth [10].

THEOREM 2: *The average case of the dichotomic algorithm satisfies the assymptotic bound,*

$$\lim_{n\to\infty} \frac{\overline{\ell(n, \sigma)}}{\lambda(n)} \approx \frac{1}{2} + \frac{6 \ln^2 2}{\pi^2} \left[1 + \sum_{r=2}^{\infty} P(r) \cdot \ell(r, \sigma)\right]. \tag{3}$$

Proof: Let k be an m-digits integer (e.g. $\lambda(k) + 1 = m$). Let $q = 2^m$ and let $r \in [0, k[$ be a random variable. Then the average value of $\ell(n, \sigma)$ for $n = qk + r$ is

$$\overline{\ell(n, \sigma)} = \frac{1}{k} \sum_{r=1}^{k} \ell(qk + r, \sigma), \qquad \text{for} \quad n \in [qk, (q+1)k - 1[.$$

By definition of $\ell(n, \sigma)$ we have

$$\overline{\ell(n, \sigma)} = \frac{1}{k} \sum_{r=1}^{k-1} (\ell(q, \sigma) + 1 + L(k, r)) = \ell(q, \sigma) + 1 + \frac{1}{k} \sum_{r=1}^{k-1} L(k, r).$$

Following Knuth [10], let $T(k, r)$ be the length of the continued fraction of k/r. Since $\ell(q, \sigma) = m$ it follows that

$$\overline{\ell(n, \sigma)} = m + 1 + \frac{1}{k} \sum_{r=1}^{k-1} \left(\ell(Gcd(k, r), \sigma) + T(k, r) - 1 + \sum_{i=1}^{T(k,r)} \ell(q_i(k, r), \sigma)\right)$$

where $q_i(k, r)$ is the i-th partial quotient of k/r. Then,

$$\overline{\ell(n, \sigma)} = m + \frac{1}{k} \sum_{r=1}^{k-1} T(k, r) + \frac{1}{k} \sum_{r=1}^{k-1} \left[\ell(Gcd(k, r), \sigma) + \sum_{i=1}^{T(k,r)} \ell(q_i(k, r), \sigma)\right].$$

Let $\overline{T(k)}$ be the average length of the continued fractions when r ranges in $[1, k-1]$. Then, the expression between brackets is summed over all $k \cdot \overline{T(k)}$ partial quotients appearing in the continued fractions. Therefore,

$$\overline{\ell(n, \sigma)} \approx m + \overline{T(k)} + \frac{1}{k} \sum_{r=2}^{k-1} P(r) \cdot k \cdot \overline{T(k)} \cdot \ell(r, \sigma),$$

$$\approx m + \overline{T(k)} + \overline{T(k)} \sum_{r=2}^{k-1} P(r) \cdot \ell(r, \sigma).$$

But $\overline{T(k)} \approx 12 \ln(2)/\pi^2 \ln(k)$ and $\ln(k) = \ln(2) \log_2(k)$ so that,

$$\overline{\ell(n, \sigma)} \approx \lambda(n) \left(\frac{1}{2} + \frac{6 \ln^2 2}{\pi^2} \left[1 + \sum_{r=2}^{k-1} P(r) \cdot \ell(r, \sigma) \right] \right).$$

Taking the limit when $k \to \infty$, we obtain the desired result. ■

The probability $P(r)$ in the theorem is estimated by (*see* [10])

$$P(r) \approx \log_2 \left(\frac{(r+1)^2}{r(r+2)} \right) = \log_2 \left(1 + \frac{1}{r(r+2)} \right).$$

Therefore, the series in (3) is approximately

$$\sum_{r=2}^{\infty} P(r) \cdot \ell(r, \sigma) \approx \frac{1}{\ln(2)} \sum_{r=2}^{\infty} \ln \left(1 + \frac{1}{r(r+2)} \right) \cdot \ell(r, \sigma)$$

$$< \frac{1.5}{\ln^2(2)} \sum_{r=2}^{\infty} \ln \left(1 + \frac{1}{r(r+2)} \right) \cdot \ln(r). \quad (4)$$

The asymptotic expansion (computed with MAPLE) of the summand in the series above (4) is

$$\ln \left(1 + \frac{1}{r(r+2)} \right) \ln(r)$$

$$\approx \frac{\ln(r)}{r^2} - \frac{2 \ln(r)}{r^3} + \frac{7 \ln(r)}{2 r^4} - \frac{6 \ln(r)}{r^5} + O(r^{-6}).$$

It shows that the series is convergent. Now, the computation of the first 1000 elements shows that $\overline{\ell(n, \sigma)} \approx (1.294508882\ldots) \lambda(n)$. The error ε is computed as follows

$$\varepsilon < \frac{6 \ln^2 2}{\pi^2} \cdot \frac{1.5}{\ln^2(2)} \int_{r=10001}^{\infty} \ln \left(1 + \frac{1}{r(r+2)} \right) \cdot \ln(r),$$

which yields, after having computed the integral with MAPLE,

$$\varepsilon < (.2920804082)\,(3.122053472)\,(.001020844836) \approx .0009308988636.$$

3.3. Space-Time tradeoff

Indeed, the effective construction of an optimal euclidean chain is equivalent to the computation of its associated dag. More precisely, addition chains are represented by linked lists. Each node represents one term of the chain, and two pointers link this node to the terms whose sum give this term. Figure 1 gives an example of such a representation for the chain (1, 2, 4, 5, 10, 20, 40, 80, 85, 87).

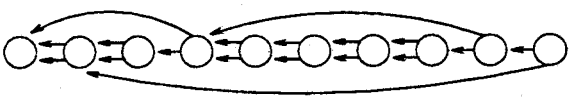

Figure 1. – Representation.

Observe that the terms of the chain are not part of this representation. This allows for a constant time implementation of the basis operations $\otimes$ and $\oplus$. This particular aspect of the problem was studied in ([7], [8]) using a Scheme implementation. A careful analysis of the recursive calls shows that, at each level of recursion we need to remember the values of at most two numbers. It follows that the number of registers needed to compute the dag corresponding to n is $\mathbf{O}(\lambda(\lambda(n)))$.

In comparison the multiplication scheme for n associated with the binary method is given by its binary representation, and is basically free. However the computation time of an euclidean chain using the dichotomic principle is low enough to consider it as a good alternative to the binary algorithm (*cf.* Example 4).

Example 4: The computation time required for constructing 1,000,000 chains for numbers in the range $[2^{30}, 2^{31} - 1]$ required 5:40 minutes of CPU time. Therefore the time required for the computation of one addition chain is about 0.340 ms. If one restricts to the construction of the dag associated, the time required will be reduced accordingly since the time shown includes the effective computation of the addition chain.

3.4. Remarks

The algorithm can be improved as follows. First observe that every chain contains 2. From this fact one can ask if it is suitable to compute $Euclid(n, 2)$. Computer calculations summarized in Table 4 show that $L(n, 2) > \ell(n, \sigma)$ in 30% of the cases while the equality holds in 55% of them.

TABLE 4
Comparison between $L(n, 2)$ *and* $\ell(n, \sigma)$ *for* $2 \leq n \leq 8192$.

Comparison	Cases	Gain
$L(n, 2) > \ell(n, \sigma)$	2646	–2 983
$L(n, 2) = \ell(n, \sigma)$	4509	0
$L(n, 2) < \ell(n, \sigma)$	1158	1276

So, when using $Chain(n)$ instead of $Euclid\ (n, 2)$, the total gain is 1707, and to force the computation of $Chain(n)$ in that case, we need to insert in algorithm *Chain* the line

$$elif\ r \leq 2\ then\ return\ Chain(n_2) \otimes Chain(q) \oplus r \qquad (\dagger)$$

Another improvement arises from the fact that the binary method is optimal for all n such that $\nu(n) \leq 3$. Using this fact one can replace the first two lines in algorithm *Chain* by

$$if\ \nu(n) \leq 3\ then\ return,\ Binary\ Chain(n) \qquad (\ddagger)$$

We suppose here that one has available a function *Binary Chain* (n) which returns a chain for n according to the standard binary method.

In Table 5 we compare the dichotomic methods deduced from the observations above. *Dichotomic* † stands for the algorithm deduced by forcing the computation of $Chain(n)$ instead of $Euclid(n, 2)$ (†), and *Dichotomic* ‡ stands for the additional modification (‡) which forces the computatin of the binary method when $\nu(n) \leq 3$.

TABLE 5
Comparison of the dichotomic strategies for $2 \leq n \leq 8192$.

Strategy	Total length	Difference
Binary	135184	0
Dichotomic	122997	12187
Dichotomic †	122896	12288
Dichotomic ‡	122859	12325

4. CONCLUSION

The euclidan dichotomic algorithm is much better than the binary algorithm, but it is not optimal. Extensive computations reveal, as expected, that it is better than Thurber's asymptotically optimal algorithm, for numbers having less than 80 digits, and suggest the use of an algorithm which adapts according to the length of the binary representation of numbers.

The number of registers required for producing a dag associated with a given number n is $\mathbf{O}(\lambda(\lambda(n)))$. Since the depth of the dag is exactly the length of the addition chain, there is clearly no data compression when using dags instead of the standard binary representation. However, it improves drastically the exponentiation process, especially when working with semigroups having a costly multiplication: matrix multiplication is such an operation; euclidean chains also form a semigroup with product $\otimes$.

REFERENCES

1. A. Brauer, On Addition Chains, *Bull. Amer. Math. Soc.*, 1939, *45*, pp. 736-739.
2. F. Bergeron, J. Berstel and S. Brlek, A Unifying Approach to the Generation of Addition Chains, *Proc. XV Latin American Conf. on Informatics, Santiago, Chile,* July 10-14, 1989, pp. 29-38.
3. F. Bergeron, J. Berstel and S. Brlek, Efficient Computation of Addition Chains, *J. Théorie des Nombres de Bordeaux*, 1994, *6*, pp. 21-38.
4. F. Bergeron, J. Berstel, S. Brlek and C. Duboc, Addition Chains Using Continued Fractions, *J. Algorithms,* 1989, *10*, pp. 403-412.
5. F. Bergeron and J. Olivos, Vectorial Addition Chains Using Euclid's Algorithm, Submitted.
6. J. Bos and M. Coster, Addition Chain Heuristics, Proceedings of CRYPTO 89, 1989. – Some algorithms on addition chains and their complexity *Tech. Rep. Center for Mathematics and Computer Sc. Amsterdam,* 1990.
7. S. Brlek, P. Castéran and R. Strandh, Chaînes d'additions et structures de contrôle, *Journées JFLA 91, 28-29 janvier 1991, Gresse-en-Vercors, France, Bigre*, 1991, *72*, pp. 54-63.
8. S. Brlek, P. Castéran and R. Strandh, On Addition Schemes, Proceedings of TAPSOFT91, April 8-12, 1991, Brighton, England, *Lect. Notes in Comp. Sci.*, 1991, *494*, pp. 379-393.
9. P. Downey, B. Leong and R. Sethi, Computing Sequences with Addition Chains, *SIAM J. Computing,* 1981, *10*, pp. 638-646.
10. D. E. Knuth, *The Art of Computer Programming*, vol. 2, Addison-Wesley, 1981.
11. F. Morain and J. Olivos, Speeding up the computations on an elliptic curve using addition-substraction chains, *RAIRO Theo. Informatics and Appl.,* 1990, 6, *24*, pp. 531-543.

12. A. SCHÖNHAGE, A Lower Bound for the Length of Addition Chains, *Theoretical Comp. Sci.*, 1975, *1*, pp. 1-12.
13. E. G. THURBER, On Addition Chains $\ell(mn) \leq \ell(n) - b$ and Lower Bounds for $c(r)$, *Duke Math., J.*, 1973, *40*, pp. 907-913.
14. A. C.-C. YAO, On the Evaluation of Powers, *SIAM J. Comp.*, 1976, *9*, pp. 100-103.