Worst case analysis of Poincaré-like algorithms

Sidi Mohamed SEDJELMACI

LIPN CNRS UMR 7030,
Université Paris-Nord
Av. J.-B. Clément, 93430 Villetaneuse, France.
E-mail: sms @lipn.univ-paris13.fr
Version: March 6th 2021.

Abstract: We prove the following result : Let n be a large integer. Let p_{max} be the largest number of iterations in a Poincaré-like algorithm for an input vector A of $n+1$ integers of at most n bits in size, Then

$$p_{max} = \lfloor\, \lambda_0\, n + \alpha\, \ln(n) + \beta\, \rfloor\,, \quad \text{where}$$

$$\alpha = \frac{1}{2\,\Phi'(\lambda_0)} \;\text{ and }\; \beta = \frac{\ln(\sqrt{2\pi})}{\Phi'(\lambda_0)} + \frac{1}{2}\,,$$

where $\Phi(\lambda) = (1+\lambda)\ln(1+\lambda) - \lambda\ln(\lambda)$ for $\frac{1}{8} < \lambda < \frac{1}{2}$, and $\lambda_0 \simeq 0.2938$ is the solution of $\Phi(\lambda) = \ln(2)$.

Keywords: Poincaré algorithm; Complexity analysis; Number theory.

---

1 Introduction

This paper deals with the complexity analysis of Poincaré-like algorithms that differ only on their exit test. More precisely, we only consider a simple case of these related algorithms and study its worst case. The other cases could be inspired by this basic case.

We prove our result in two times. First we prove that (Proposition 1) if p and N are two positive integers and $A = (a_1, a_2, \cdots, a_N)$ is any input vector of positive integers, then the Poincaré algorithm reaches $A = (1, 1, \cdots, 1)$ after p iterations with

$$\max(A) \geq \tbinom{p+N-1}{N-1}\,.$$

Then, for the case $N = n+1$ (Theorem 1 and Corrolary 2), we give the exact value of p_{max}, the maximun number of iterations in Poincaré's algorithm.

In order to be self-contained, we start with the following:

Notation and definitions: Let $A = (a_1, a_2, \cdots, a_N)$ and $B = (b_1, b_2, \cdots, n_N)$ be two vectors of N distinct integers, $N \geq 4$.

- $\max(A) = \max\{a_1, a_2, \cdots, a_N\}$; $\min(A) = \min\{a_1, a_2, \cdots, a_N\}$.
- $A \leq B$ if and only if $\forall i\,,\ 1 \leq i \leq N,\ a_i \leq b_j$.
- $\Delta\, A = (a_1, a_2 - a_1, \cdots a_{i+1} - a_i, \cdots, a_N - a_{N-1})$.

- $\Delta^{-1}\,A = (a_1\,,\,a_1+a_2\,,\cdots,a_1+a_2+\cdots+a_i\,,\cdots,\,a_1+\cdots a_N)$.
- Δ^{-1} satisfies $\Delta\,(\Delta^{-1}A) = A$ and $A \leq B \Longrightarrow \Delta^{-1}(A) \leq \Delta^{-1}(B)$.
- $A_+ = \mathrm{Sort}(A)$ is the increasing sorted vector of A.
- $\mathcal{P}_N$ is the set of all one to one and onto applications from $(1,2,\cdots,N)$ to $(1,2,\cdots,N)$ (also called permutations), $Id \in \mathcal{P}_N$ is the identity permutation: $Id(A) = A$.
- $\forall \sigma \in \mathcal{P}_N$, $\sigma(A) = (a_{\sigma(1)}, a_{\sigma(2)}, \cdots, a_{\sigma(N)})$.
- σ_{max} is defined by $\sigma_{max}(A) = (a_{\sigma(1)}, \cdots, a_{\sigma(N)})$, with $a_{\sigma(i)} > a_{\sigma(i+1)}$, $1 \leq i \leq N-1$.
- Let $(u_n)_n$ be sequence of reals, then $(u_n)_n = o(1)$, means $(u_n) \to 0$, when $n \to +\infty$.

Example 1 : $(N = 5)$
$A = (22, 35, 51, 84, 113)$. So $\Delta A = (22, 13, 16, 33, 29)$ and $(\Delta A)_+ = (13, 16, 22, 29, 33)$,
$\sigma_{max}(A) = (113, 84, 51, 35, 22)$ and $\Delta^{-1}(\sigma_{max}(A)) = (113, 197, 248, 283, 305)$.

2 The algorithm

We may repeat the same process $A \leftarrow (\Delta A)_+$ to the vector $(\Delta A)_+$ instead of A and so on until we reach a vector with small enough integers. Note also that $(\Delta A)_+ = M \times A$, where M is an unimodular matrix, i.e.: $\det(M) = \pm 1$. The following algorithm due to Poincaré [3] shows how to reduce the size of integers of A without divisions and, in the same time, preserving some properties. Actually the author used it to compute some volumes. It was also used for computing the GCD of many integers in parallel [2].

A Poincaré-like algorithm : $\mathcal{A}$

Input: $A = (a_1, \cdots, a_N)$ is a vector of N positive distinct integers, such that $\max\{A\} < 2^n$; $n, N \geq 4$ and a boolean condition **cond** on A such that **cond**(**A**) is truth in the beginning.

Outut: $A = (a_1, \cdots, a_N)$ such that **cond**(**A**) is false.

```
A := Sort(A) ;
While (cond(A)) Do
    A := Δ(A) ;
    A := Sort(A) ;   /* In increasing order */
Endwhile
Return A ;
```

In this paper we are concerned with a week condition on A: $\min(A) > 0$ i.e.:
While $(\min(A) > 0)$ **Do**. It also means that we stop when we reach $a_i = 0$, for some index $1 \leq i \leq N$. We want to find the worst case of this related algorithm.

We start by the following observations for the worst case:

1) : Since the inputs $A = (a_1, a_2, \ldots, a_N)$ and $cA = (ca_1, ca_2, \ldots, ca_N)$, for $c \geq 1$ have exactly the same number of iterations in algorithm $\mathcal{A}$, then the worst case is obtained when $\gcd(A) = \gcd(a_1, \cdots, a_N) = 1$.

2) : Let $p = p(N, n)$ be the number of iterations before $\min(A) = 0$. Then after p iterations we have $A = (1, 1, 1, \cdots, 1)$.

3) : From each step the previous vector $A = (a_1, \cdots, a_N)$ can be obtained by the operator Δ^{-1} such that $\Delta(\Delta^{-1}(A)) = A$, defined by

$$\Delta^{-1}(a_1, a_2, a_3, \cdots, a_N) = (a_1, a_1 + a_2, a_1 + a_2 + a_3, \cdots, a_1 + a_2 + \cdots a_N)\,.$$

Recall that we start the algorithm with the condition $\max(A) < 2^n$. Now if n and N are fixed, then we want to now what is the maximun number of iterations $p_1 = p(N, n)$ to reach the vector $A = (1, 1, \cdots, 1)$. The algorithm stops in the next step ($\min(A) = 0$), and the maximal number of steps in the algorithm is $p_{max} = p_1 + 1$. We focus on the value of p_1.

Let $A^{(k)}$ be the value of vector A at iteration k, $k \geq 0$, with $A^{(0)} = A$. Each iteration expresses as $A^{(k+1)} = \Delta(A^{(k)})_+ = (\pi \circ \Delta)A^{(k)}$, for some permutation $\pi \in \mathcal{P}_N$ depanding on $\Delta(A^{(k)})$ such that $(\pi \circ \Delta)A^{(k)}$ is increasingly sorted.

Suppose that at iteration p we reach $A^{(p)} = (1, 1, \cdots, 1)$. We observe that $\Delta^{-1}(A^{(p)}) = \Delta^{-1}(1, 1, \cdots, 1) = (1, 2, \cdots, N)$. Since $A^{(k+1)} = (\pi \circ \Delta)A^{(k)}$ for $k \geq 1$, so for $k = p - 1$, $A^{(p)} = (\pi \circ \Delta)\, A^{(p-1)}$ and

$$A^{(p-1)} = (\Delta^{-1} \circ \pi^{-1})\, A^{(p)}\,.$$

Here we cannot compute $A^{(p-1)}$ because we dont know the permutation π^{-1}. In the following we consider a special case.

Lemma 1: Let $0 \leq k \leq p$. If we assume that $A^{(p)} = (1, 1, \cdots, 1)$ and for each iteration k, π is always the identity $\pi = Id$, i.e.: $Id(A^{(k)}) = A^{(k)}$. Then at iteration $p - k$, we have

$$A^{(p-k)} = (\tbinom{k+1}{1}, \tbinom{k+2}{2}, \tbinom{k+3}{3}, \cdots, \tbinom{k+N-1}{N-1})) \qquad (1)\,.$$

Proof: Using Δ^{-1} each time, we have in turn :
$A^{(p)} = (1, 1, \cdots, 1)$.
$A^{(p-1)} = (1, 2, 3, 4, 5, \cdots, N-1, N)$.
$A^{(p-2)} = (1, 3, 6, 10, 15 \cdots, N(N+1)/2)$.
So the property is valid for $k = 0, 1, 2$. By induction. Suppose that for a given $1 \leq k \leq N - 1$, the relation (1) is valid for k. i.e.: $A^{(p-k)} = (1, \tbinom{k+1}{1}, \tbinom{k+2}{2}, \tbinom{k+3}{3}, \cdots, \tbinom{k+N-1}{N-1}))$. Let $A^{(p-k)} = (a_1^{(p-k)}, a_2^{(p-k)}, a_3^{(p-k)}, a_N^{(p-k)})$. So, for some k and $1 \leq j \leq N$, we have

$$a_j^{(p-k)} = \tbinom{k+j-1}{j-1} \qquad (2)\,.$$

Recall that $\Delta^{-1}(x_1, x_2, \cdots, x_N) = (y_1, y_2, \cdots, y_N)$ with $y_j = \sum_{i=1}^{j} x_i$.
Since $A^{(p-k-1)} = (a_1^{(p-k-1)}, a_2^{(p-k-1)}, \cdots, a_N^{(p-k-1)}) = \Delta^{-1}(A^{(p-k)})$ then

$$a_j^{(p-k-1)} = \sum_{i=1}^{j} a_i^{(p-k)} = \sum_{i=1}^{j} \tbinom{k+i-1}{i-1} = \tbinom{k+j}{j-1}\,. \qquad \square$$

Proposition 1: Let p and N be two a positive integers and $A = (a_1, a_2, \cdots, a_N)$ be any input vector of positive integers. If the algorithm reach $A = (1, 1, \cdots, 1)$ after p iterations then

$$\max(A) \geq \tbinom{p+N-1}{N-1}\,.$$

Proof: Let $A = (a_1, \cdots, a_N)$ is a vector of N distinct integers, such that $0 < a_1 < a_2 < \cdots < a_{N-1} < a_N$ and $\sigma \in \mathcal{P}_N$. Recall that $\sigma(A) = A = (a_{\sigma(1)}, a_{\sigma(1)}, \cdots, a_{\sigma(N)})$.
Let $\Delta^{-1}(A) = B = (b_1, b_2, \cdots, b_N)$, with $b_j = \sum_{i=1}^{j} a_i$ and
$\Delta^{-1}(\sigma(A)) = C = (c_1, c_2, \cdots, c_N)$, with $c_j = \sum_{i=1}^{j} a_{\sigma(i)}$. Since $a_{\sigma(j)} \in A$, for all $1 \leq k \leq N$, so c_j is a sum of j distinct integers belonging to A. But b_j is the smallest sum of j distinct integers belonging to A which are $a_1, a_2, \cdots, a_j$, so $b_j \leq c_j$ or

$$\forall \sigma \in \mathcal{P}_N\,, \qquad \Delta^{-1}(A) = \Delta^{-1}(Id(A)) \leq \Delta^{-1}(\sigma(A)) \leq \Delta^{-1}(\sigma_{max}(A))\,,$$

where $\sigma_{max}(A)$ is the decreasing sorted vector of A. So

$$\forall \pi^{-1} \in \mathcal{P}_N\,,\ \forall k\,, 1 \leq k \leq p\,, \qquad \Delta^{-1}(Id(A^{(p-k)})) \leq \Delta^{-1}(\,\pi^{-1}(A^{(p-k)})\,)\,.$$

At each step k in the left side, only the identity $Id \in \mathcal{P}_N$ is considered and Lemma 1 yields for $k = p$, $\max(A) = \binom{p+N-1}{N-1}$. For the right side we consider, at each step k, any permutation $\pi^{-1} \in \mathcal{P}_N$ so $\max(A) \geq \binom{p+N-1}{N-1}$. □

Corollary 1: Let p and N be two a positive integers and $A = (a_1, a_2, \cdots, a_N)$ be any input vector of positive integers. If

$$\binom{p+N-1}{N-1} \leq \max(A) < \binom{p+N}{N-1}\,,$$

then the algorithm terminates after at most $p+1$ iterations.

Proof: Thanks to Proposition 1, the worst case to obtain the last but one iteration is p, so the worst case to reach the condition $\min\{A\} = 0$ is $p+1$.

Example 2: If the number of integers of an input vector A is $N = 5$ and if $\max\{A\} = 113$, then $p = 4$ since the sequence $\left(\binom{k+N-1}{N-1}\right)_{k\geq 0}$ yields

$$70 = \binom{8}{4} = \binom{p+N-1}{N-1} < 113 < 126 = \binom{9}{4} = \binom{p+N}{N-1}\,,$$

and for any such vector A, the algorithm terminates for at most $p+1 = 5$ iterations. More precisely, for any vector A of 5 positive integers such that $70 \leq \max\{A\} < 126$, the worst case is given by the vector

$$\left(\binom{p}{0}, \binom{p+1}{1}, \binom{p+2}{2}, \binom{p+3}{3}, \cdots, \binom{p+N-1}{N-1}\right) = (\,1, 5, 15, 35, 70\,)\,.$$

The aim of the following is to give a closed form of $p_1 = p_1(N, n)$, the maximum of iterations to reach the vector $(1, 1, \cdots, 1)$, so that $p_{max} = p_1(N, n) + 1$.

Lemma 2: Let N, p be two large positive integers and x a real, such that $1 \leq x$, $p \leq N$, then

$$1) \quad f(x) = f_N(x) = \frac{(N+x)^{N+x+1/2}}{x^{x+1/2}} \quad \text{is increasing}\,.$$

$$2) \qquad \binom{p+N}{N} = \left(\frac{1}{\sqrt{2\pi}} + o(1)\right) \frac{f(p)}{N^{N+1/2}}\,.$$

Proof: 1) We observe that $f(x) > 0$. Let $F(x) = \ln f(x) = (N + x + 1/2)\ln(N + x) - (x + 1/2)\ln(x)$ so that $f(x) = e^{F(x)}$ and $f'(x) = F'(x)e^{F(x)}$. Now

$$F'(x) = \ln\left(\frac{N+x}{x}\right) - \frac{N}{2x(N+x)} \quad \text{and}$$

$$F''(x) = \frac{-N(\,x^2 + (N-1)x - (N/2)\,)}{(Nx + x^2)^2}\,.$$

Since $x^2 + (N-1)x - (N/2) > 0$ for $1 \le x \le N$, then $F''(x) < 0$, F' is decreasing and $F'(x) \ge F'(N) = \ln 2 - (1/4N) > 0$. Hence F and f are both increasing.
2) We assume that both n and p are all tending to infinity, so we can use Stirling's approximation to evaluate all the factorials. We obtain

$$\begin{aligned}
\binom{p+N}{N} &= (1 + o(1)) \sqrt{\frac{N+p}{2\pi pN}}\ \frac{(N+p)^{N+p}}{p^p N^N} \\
&= \left(\frac{1}{\sqrt{2\pi}} + o(1)\right)\ \frac{(N+p)^{N+p+1/2}}{p^{p+1/2} N^{N+1/2}}\,.
\end{aligned}$$

Hence the result. □

Recall that p_1 be the largest number of iterations in the algorithm to reach the vector $(1, 1, \cdots, 1)$ for an input vector A of N integers of at most n bits in lenght. From Proposition 1, we derive

$$p_1 = \max\{\, 1 \le p \le N-1\,, \quad \text{s.t.}: \quad \binom{p+N-1}{N-1} < 2^n \,\}\,.$$

From a parallel algorithmic point of view, we must have $N = n^{O(1)}$, i.e.: polynomial with respect to n, and for the sake of readability, from now on, we fix $N = n + 1$ so

$$p_1 = p_1(n) = \max\{\, 1 \le p \le n\,, \quad \text{s.t.}: \quad \binom{p+n}{n} < 2^n \,\}\,.$$

Moreover, Proposition 1 and Lemma 2 yield

$$p_1 = \max\{\, 1 \le p \le n\,, \quad \text{s.t.}: \quad f(p) < \{\sqrt{2\pi} + o(1)\} \times 2^n\, n^{n+1/2} \,\}\,, \tag{3}$$

where

$$f(x) = f_n(x) = \frac{(n+x)^{n+x+1/2}}{x^{x+1/2}}\,.$$

Lemma 3 : If $N = n+1$ and $p_1 = \max\{\, 1 \le p \le n\,,\ \text{s.t.}: \binom{p+n}{n} < 2^n \,\}$ and let $\lambda = p_1/n$, then

$$i) \quad \lfloor \frac{n}{4} \rfloor \le p_1 < \lceil \frac{n}{2} \rceil\,.$$

$$ii) \quad \forall n \ge 8\,, \qquad \frac{1}{8} \le \lambda < \frac{1}{2}\,.$$

Proof: First we prove that $p < n$, then $n/4 < p < n/2$.

$$f(n) = \frac{(2n)^{2n+1/2}}{n^{n+1/2}} = 2^{2n+1/2} n^n > (\sqrt{2\pi} + o(1)\,)\, 2^n n^{n+1/2}\,.$$

Since f is increasing so we must have $p < n$.

If $\lambda = p/n$, we observe that

$$f(p) = f(\lambda n) = \sqrt{\frac{1+\lambda}{\lambda}}\ n^n\ \{\, (\frac{1+\lambda}{\lambda})^{\lambda}\ (1+\lambda)\,\}^n\,.$$

• If $p = n/2$, then $\lambda = 1/2$ and $3\sqrt{3} > 4$, so for $n \geq 4$

$$f(n/2) = \sqrt{3}\, n^n \, (\frac{3\sqrt{3}}{2})^n \; > (\sqrt{2\pi} + o(1))\, (2^n n^{n+1/2})\,.$$

Hence

$$f(\lceil n/2 \rceil) \geq f(n/2) > (\sqrt{2\pi} + o(1)\,)\, (2^n n^{n+1/2})\,,$$

and by (3), $p_1 < \lceil n/2 \rceil$, or $p_1 \leq \lceil n/2 \rceil - 1 < n/2$.

• If $p = n/4$, then, since $5 \times 5^{1/4} < 8$:

$$f(\frac{n}{4}) = \sqrt{5}\; n^n \; (\frac{5 \times 5^{1/4}}{4}\,)^n \; < (\,\sqrt{2\pi} + o(1)\;)\, (2^n n^{n+1/2})\,.$$

So by (3), $\lfloor n/4 \rfloor \leq p_1$ and $p_1 \geq n/4 - 1$, hence $i)$.

$ii)$ Let $\lambda = p_1/n$ then, from $i)$ and for $n \geq 8$:

$$\frac{1}{8} \leq \frac{1}{4} - \frac{1}{n} \leq \lambda = \frac{p_1}{n} < \frac{1}{2}\,. \qquad \square$$

Lemma 4: Let $\Phi(x) = (1+x)\ln(1+x) - x \ln x$ for $1/8 < x < 1/2$.

If $N = n + 1$ and $p = \lambda n$ with $1/8 < \lambda < 1/2$, then

$$f(\lambda n) \leq (\sqrt{2\pi} + o(1)\,)\; n^{n+1/2} 2^n \iff \Phi(\lambda) \leq \ln 2 + \frac{\ln(n) - \Phi'(\lambda)}{2n} + \frac{\ln(\,\sqrt{2\pi} + o(1)\,)}{n}\,.$$

Proof: Let $p = \lambda n$ with $1/8 < \lambda < 1/2$, then

$$f(p) = f(\lambda n) = n^n \, \frac{(1+\lambda)^{n+\lambda n+1/2}}{\lambda^{\lambda n+1/2}}\,.$$

So

$$\ln(f(p)) = \ln(f(\lambda n)) = n \ln n + (n + \lambda n + 1/2) \ln(1+\lambda) - (\lambda n + 1/2)\; \ln \lambda\,.$$

Then $f(p) = f(\lambda n) \leq (\sqrt{2\pi} + o(1)\,)\; n^{n+1/2} 2^n$ yields

$$n\{\,(1+\lambda)\ln(1+\lambda) - \lambda \ln(\lambda) - \ln(2)\,\} \leq \frac{1}{2} \ln(\frac{\lambda n}{1+\lambda}) + \ln(\sqrt{2\pi}) \; + o(1)\,,$$

and

$$(1+\lambda)\ln(1+\lambda) - \lambda \ln(\lambda) \leq \ln(2) + \frac{1}{n}\,\{\,\frac{1}{2} \ln(\frac{\lambda n}{1+\lambda}) + \ln(\sqrt{2\pi}) \; + o(1)\,\}\,.$$

But

$$\ln(\frac{\lambda n}{1+\lambda}) = \ln(n) - \ln(\frac{1+\lambda}{\lambda}) = \ln(n) - \Phi'(\lambda)\,.$$

Hence the result. $\square$

In the following we consider the real λ_{max}, the maximun value of λ in Lemma 4.

Theorem 1: Let $\Phi(\lambda) = (1+\lambda)\ln(1+\lambda) - \lambda \ln(\lambda)$ for $\lambda_0 \leq \lambda < \frac{1}{2}$, where $\lambda_0 \simeq 0.2938$ is the solution of $\Phi(x) = \ln(2)$, so that $\Phi'(\lambda_0) = \ln(\frac{1+\lambda_0}{\lambda_0}) \simeq 1.4824$. Let p_1 be the largest

number of iterations to reach $A = (1, 1, \cdots, 1)$ in the algorithm $\mathcal{A}$, for an input vector A of $n+1$ integers of at most n bits in size. Then

$$p_1 = \lfloor\, \lambda_0\, n + a\, \ln(n) + \, b\, \rfloor\,, \quad \text{where}$$

$$a = \frac{1}{2\,\Phi'(\lambda_0)} \;\text{ and }\; b = \frac{\ln(\sqrt{2\pi})}{\Phi'(\lambda_0)} - \frac{1}{2}\,.$$

Proof: From Lemma 3, we have

$$f(\frac{n}{4}) < (\sqrt{2\pi} + o(1))\, n^n 2^{n+1/2} < f(\frac{n}{2})\,,$$

and since f is continuous, there exists λ_{max} and $p'_1 = \lambda_{max}\, n$ two reals such that

$$f(p'_1) = f(\lambda_{max}\, n) = (\sqrt{2\pi} + o(1))\, n^n 2^{n+1/2} \qquad (4)\,.$$

Note that λ_{max} will be defined up to $o(1/n)$ and p'_1 up to $o(1)$.

Thanks to Lemma 4, λ_{max} is the solution of the equation

$$\Phi(\lambda) = \ln(2) + \frac{\ln(n)}{2n} - \frac{\Phi'(\lambda)}{2n} + \frac{\ln(\sqrt{2\pi}) + \, o(1)}{n}\,,$$

or

$$\Phi(\lambda) + \frac{\Phi'(\lambda)}{2n} = \ln(2) + \frac{\ln(n)}{2n} + \frac{\ln(\sqrt{2\pi}) + \, o(1)}{n} \qquad (5)\,.$$

Recall that $\ln 2 = \Phi(\lambda_0)$. This suggests to seek a solution as

$$\lambda_{max} = \lambda_0 + \frac{a\ln(n) + b}{n} + o(\frac{1}{n})\,,$$

where a, b are two reals we have to find.

The function $T(x) = \Phi(x) + \frac{1}{2n}\,\Phi'(x)$ is regular for $1/8 < x < 1/2$.
For the sake of readability, here λ means λ_{max}. Taylor's theorem for fonction T yields

$$\Phi(\lambda) + \frac{\Phi'(\lambda)}{2n} = \Phi(\lambda_0) + \frac{\Phi'(\lambda_0)}{2n} + \{\, \frac{a\ln(n) + b}{n} + o(\frac{1}{n})\,\}\,\{\Phi'(\lambda_0) + \frac{\Phi''(\lambda_0)}{2n}\,\}$$

$$+\frac{1}{2}\,\{\, \frac{a\ln(n) + b}{n} + o(\frac{1}{n})\,\}^2\,\{\Phi''(t) + \frac{\Phi'''(t)}{2n}\}\,,$$

for some real t, $\lambda_0 < t < \lambda_{max}$.

However $\Phi''(t)$ and $\Phi'''(t)$ are both bounded, so the last term is an $o(1/n)$ and for $\lambda = \lambda_{max}$:

$$\Phi(\lambda) + \frac{\Phi'(\lambda)}{2n} = \Phi(\lambda_0) + \frac{\Phi'(\lambda_0)}{2n} + \{\, \frac{a\ln(n) + b}{n} + o(\frac{1}{n})\,\}\,\{\Phi'(\lambda_0) + \frac{\Phi''(\lambda_0)}{2n}\,\} + \, o(\frac{1}{n})\,.$$

$$\text{or}\quad \Phi(\lambda) + \frac{\Phi'(\lambda)}{2n} = \Phi(\lambda_0) + \{\, \frac{1}{2} + b\,\} \times \frac{\Phi'(\lambda_0)}{n} + a\,\Phi'(\lambda_0) \times \frac{\ln(n)}{n} + \, o(\frac{1}{n}) \qquad (6)\,.$$

Then (5) and (6) yield

$$a\,\Phi'(\lambda_0) = \frac{1}{2} \;\text{ and }\; (\frac{1}{2} + b)\,\Phi'(\lambda_0) = \ln(\sqrt{2\pi})\,.$$

n	10	11	12	13	14	15	16	17	20	25
$p_1(n)$	4	4	4	4	5	5	5	6	7	8

Table 1: Some elements of the sequence $(p_1(n))_{n\geq 1}$.

Hence

$$a = \frac{1}{2\,\Phi'(\lambda_0)} \quad \text{and} \quad b = \frac{\ln(\sqrt{2\pi})}{\Phi'(\lambda_0)} - \frac{1}{2} \qquad (7)\,.$$

Corrolary 2 : The maximal number of iteration in algorithm $\mathcal{A}$ for an input vector of $n+1$ integers of at most n bits in size is

$$p_{max} = \lfloor \lambda_{max} n + 1 \rfloor = \lfloor \lambda_0 n + a\ln(n) + b + 1 \rfloor\,,$$

for large enough integers n, and where a and b are defined in (7). Moreover, p_{max} is obtained for the input vector

$$A = (\,1, \tbinom{p+1}{1}, \tbinom{p+2}{2}, \tbinom{p+3}{3}, \cdots, \tbinom{p+n}{p}\,)\,.$$

Proof: From (3) and (4), we derive $f(p_1) < (\sqrt{2\pi} + o(1))\, n^n 2^{n+1/2} = f(p_1')$, where p_1' is the real defined in (4). Since f is a continuous increasing function then $p_1 < p_1' = \lambda_{max}\, n$. Recall that p_1 is an integer while p_1' is a real. Moreover, λ_0, a and b are not rationals, so

$$\lfloor \lambda_{max}\, n \rfloor < \lambda_{max}\, n = p_1' \quad \text{and} \quad f(\lfloor \lambda_{max}\, n) < f(\lambda_{max}\, n) = f(p_1')\,.$$

So the integer $\lfloor \lambda_{max}\, n \rfloor$ is the largest integer p satisfying

$$f(p) < f(p_1') = (\sqrt{2\pi} + o(1))\, n^n 2^{n+1/2}\,,$$

then by (3)

$$p_1 = \lfloor \lambda_{max} n \rfloor = \lfloor \lambda_0 n + a\ln(n) + b \rfloor\,.$$

The maximal number of iteration in the Algorithm $\mathcal{A}$ is $p_{max} = p_1 + 1$ since we must perform one more iteration to reach $\min(A) = 0$. So

$$p_{max} \leq \lfloor \lambda_{max} n \rfloor + 1 = \lfloor \lambda_0 n + a\ln(n) + b + o(1) + 1 \rfloor = \lfloor \lambda_0 n + a\ln(n) + b + 1 \rfloor\,,$$

where a and b defined in (7). □

Remarks :

• Actually, we have found a sequence $(p_1(n))_n$ satisfying

$$p_1(n) = \max\{\, p \mid \tbinom{n+p}{n} < 2^n \} = \lfloor \lambda_0 n + a\ln(n) + b \,\rfloor\,,$$

where a and b defined in (7). Some examples are given in Table 1.

• The formula of p_1 is valid only for large enough n, because the hidden constant behind $o(1)$ could be large.

• Numerical values of $\alpha = a$ and $\beta = b + 1$ yield approximately

$$p_{max} \simeq \lfloor\, (0.2938)\, n + 0.3373\, \ln(n) + \, 1.1198\, \rfloor\, .$$

Example 3 : For $n = 20$, we have $\binom{27}{20} = \binom{27}{7} < 2^{20} < \binom{28}{8} = \binom{28}{20}$ which yields

$$p_1 = p_1(20) = \max\{\, p \mid \, \binom{20+p}{20} < 2^{20}\} = 7\, .$$

So $p_{max} = p_1 + 1 = 8$. The worst case is given by the input vector A of 21 positive integers od at most 20 bits in size :

$$A = (\, 1, \binom{8}{1}, \binom{9}{2}, \binom{10}{3}, \binom{11}{4}, \binom{12}{5}, \ldots, \binom{25}{18}, \binom{26}{19}, \binom{27}{20}\,)$$

$$= (1, 8, 36, 120, 330, 792, \ldots, 230230, 888030)\, .$$

We easily check that after 8 iterations we obtain $\min\{A\} = 0$. Moreover, for $n = 20$:

$$\lfloor\, (0.2938)\, n + 0.3373\, \ln(n) + \, 1.1198 \rfloor = \lfloor\, 8.00626 \cdots \rfloor = 8\, .$$

References

[1] V. Brun, Algorithmes Euclidiens pour trois et quatre nombres, *13th Congr. Math. Scand. Helsinki, S.* (1957) 45–64.

[2] S.M. Sedjelmaci, Fast parallel GCD algorithm for many integers *Poster talk presented at ISSAC 2013, Boston, USA* and ACM Communications in Computer Algebra, Vol 47, No. 3, Issue 185, September 2013, 92-93.

[3] H. Poincaré, Sur une généralisation des fractions continues, *in C. R. Acad. Sci. Paris Sér. A 99,* (1884) 1014–1016.