DYNAMICAL ANALYSIS OF THE PARAMETERIZED LEHMER-EUCLID ALGORITHM

BENOÎT DAIREAUX, BRIGITTE VALLÉE

Abstract. The Lehmer-Euclid algorithm is an improvement of the Euclid algorithm when applied to large integers. The original Lehmer–Euclid Algorithm replaces divisions on multi–precision integers by divisions on single–precision integers. Here, we study a slightly different algorithm that replaces computations on n–bit integers by computations on μn–bit integers. This algorithm depends on the truncation degree $\mu \in]0,1[$ and is denoted as the $\mathcal{LE}_\mu$ algorithm. The original Lehmer–Euclid Algorithm can be viewed as the limit of the $\mathcal{LE}_\mu$ algorithms for $\mu \to 0$. We provide here a precise analysis of the $\mathcal{LE}_\mu$ algorithm. For this purpose, we are led to study what we call the *Interrupted Euclidean algorithm*. This algorithm depends on some parameter $\alpha \in [0,1]$ and is denoted by $\mathcal{E}_\alpha$. When running with an input (a,b), it performs the same steps as the usual Euclidean algorithm, but it stops as soon as the current integer is smaller than a^α, so that $\mathcal{E}_0$ is the classical Euclidean Algorithm. We obtain a very precise analysis of the algorithm $\mathcal{E}_\alpha$, and describe the behaviour of main parameters (number of iterations, bit-complexity) as a function of parameter α. Since the Lehmer-Euclid Algorithm $\mathcal{LE}_\mu$ when running on n–bit integers can be viewed as a sequence of executions of the Interrupted Euclidean algorithm $\mathcal{E}_{1/2}$ on μn–bit integers, we then come back to the analysis of the $\mathcal{LE}_\mu$ algorithm and obtain our results.

1. Introduction.

The computation of the GCD is a main subroutine for most of computations on long integers; it is widely used for expressing rational numbers in "lowest terms", for finding modular inverses, ... It is one of the most time-consuming basic operations on long integers. For instance, during the computation of Grobner Bases, it was noticed that 80% of computing time is spent in long integer arithmetic, and notably in GCD computations. When applied on long integers, the Euclid Algorithm is not very attractive, since it performs a sequence of multiple–precision divisions, that are quite time-consuming. Indeed, although theoretically division has the same time complexity as multiplication [17], a division algorithm that will be designed along the lines explained by Knuth will be about 30 times slower than multiplication [15]. With some improvements due to Krandick and Johnson [16], one may hope to reduce the gap to 15 times. For very long integers, Jebelean [14] proposed a division algorithm which is about two times slower than Karatsuba multiplication –in most of the cases, since there is a small probability of failure–.

A significant improvement in the speed of Euclid's algorithm when high-precision numbers are involved can be achieved with the so–called Lehmer-Euclid Algorithm that uses a method due to Lehmer [18]. Working only with the leading digits of large integers, it is possible to simulate most of the multiple-precision divisions by single–precision divisions, which leads to a significant speed-up of the algorithm. The first version of this algorithm appeared in [18]; then, some variants were described in Knuth [17], and finally, Collins [7] and Jebelean [15] provided various improvements to the algorithm. Nowadays, most of computer algebra systems and multi-precision libraries use many of these variants. However, there exist very few analyses of the Lehmer–Euclid Algorithm. Sorenson [22] obtained a worst–case analysis of this algorithm, but, to the best of our knowlegde, there does not exist any precise average-case analysis of the Lehmer–Euclid algorithm. This is the purpose of this paper to provide such an analysis.

Main results. The original Lehmer–Euclid Algorithm replaces divisions on multi–precision integers by divisions on single–precision integers (sometimes, double-precision is used). Here, we study a slightly different algorithm that replaces computations on n–bit integers by computations on μn–bit integers. This algorithm depends on the truncation degree $\mu \in]0,1[$ and is denoted as the $\mathcal{LE}_\mu$ algorithm. This Lehmer–Euclid Algorithm can be viewed as a sequence of executions of the so–called *Interrupted Euclidean Algorithm.* Generally speaking, this interrupted algorithm depends on some parameter $\alpha \in [0,1]$, and is denoted by $\mathcal{E}_\alpha$. It performs exactly the same steps as the Euclidean algorithm but, when running on some input $(a,b), a \geq b$, it stops as soon as the current integer is smaller than a^α. We first provide a complete analysis of this interrupted algorithm, and we precisely study (in the average case) its main parameters —number of iterations and bit-complexity— as a function of parameter α (Theorem 1, proved in Section 4, Theorem 2 proved in Section 5).
Then, we come back to the initial algorithm, the $\mathcal{LE}_\mu$ algorithm and we precisely compare its average bit–complexity with the average bit–complexity of the classical Euclid algorithm. We first prove that the $\mathcal{LE}_\mu$ algorithm, when running on n-bit integers, performs (almost surely) a sequence of executions of the Interrupted Euclidean Algorithm $\mathcal{E}_{1/2}$ on μn–bit integers. We then use the previous analysis of the $\mathcal{E}_\alpha$ Algorithm, and obtain a precise asymptotical value for the average bit–complexity of the $\mathcal{LE}_\mu$ algorithm when it runs on n-bit integers. This value involves, together with parameter μ, the constants M, D that intervene in the cost of a multiplication or a division, together with some constants L_1, L_2 that appear in the average bit–complexity of the classical Euclid Algorithm. More precisely, the main result of the paper is the following (Theorem 3, proved in Section 6):

When the Lehmer-Euclid algorithm deals with a truncation degree μ, its average bit-complexity on pairs of length n is asymptotically equal to

$$\left[\frac{3}{2}(L_1M + L_2D)\mu + (L_1 + L_2)M\mu + (2-\mu)M\right] n^2,$$

while the average bit-complexity of the plain Euclid Algorithm on pairs of length n is asymptotically equal to $(L_1M + L_2D)n^2$. Here, L_1 and L_2 are *the two following constants*

$$L_1 = \frac{12 \log^2 2}{\pi^2} \sim 0.58, \qquad L_2 = \frac{6 \log 2}{\pi^2} \log \prod_{k=0}^{\infty} (1 + \frac{1}{2^k}) \sim 0.66,$$

and M, D the constants that intervene in a cost of a multiplication or a division.

Here, there are some important remarks to be done. The Lehmer-Euclid algorithm $\mathcal{LE}_\mu$ is useful only in the case when a large division is more expensive than a large multiplication. More precisely, the Lehmer-Euclid algorithm may be less expensive than the Euclid algorithm only if $2M < L_1 M + L_2 D$, that is always the case on most of the computers, since one has usually $D \geq 5M$. Suppose that it is the case. Then, all the values of μ are not convenient. For instance, in the case when $D = 5M$, one has to choose a truncation degree μ at most equal to 0.3. Generally speaking, if $D/M := \rho$, the maximal value μ_0 of μ is:

$$\mu_0 = 2\,\frac{L_1 + L_2\rho - 2}{5L_1 + L_2(3\rho + 2) - 2}.$$

For $\rho = 15$, and $\mu = 1/3$, the ratio between the bit-complexity of the two algorithms (the Lehmer–Euclid Algorithm and the Euclid Algorithm) is close to 0.7. For $\rho = 30$, and $\mu = 1/10$, this ratio is close to 1/4.

Consider now a situation that may be found in the "real life". When one replaces long integers used in the RSA algorithm ($n = 1024$) by single-precision integers (32 bits), one uses a truncation degree μ equal to 1/32. Suppose also that we work with $\rho = 30$. If we apply our (asymptotical) results (only true for $n \to \infty \ldots$), we find that the ratio between the bit-complexity of the two algorithms (the Lehmer–Euclid Algorithm and the Euclid Algorithm) is now close to 1/6.

There is a general agreement between our theoretical results and the experimental curves obtained by Lercier in his thesis [19]. We reproduce these curves in Figure 1. Lercier insists on the fact that the implementation of divisions is quite different on DEC processors or on SUN processors. With our notations, the parameter ρ has thus two distinct values: its DEC-value, and its SUN-value. If it could be possible to let parameter μ tend to 0, the limit ratio between the two algorithms should be equal to

$$L(\rho) = \frac{2}{L_1 + L_2\rho}.$$

On the curves obtained by Lercier, this ratio is about 1/10 for DEC, and about 1/5 for SUN. This makes possible to obtain (somewhat!) indirect values for parameter ρ; its DEC value is near 30, and its SUN-value near 15. Moreover, for $n = 1024$, the DEC-curve exhibits a ratio of 1/12, while the SUN-curve shows a ratio about 1/6.

Methods. Most of the variants of the classical Euclidean algorithms have already been analyzed, in the worst-case, as well as in the average-case. Heilbronn and Dixon in 1969 proceeded to the first average case analysis of the Euclid algorithm. All the further average-case analyses of Euclidean algorithms are instances of what we call now a dynamical analysis. This method, due to Vallée, consists in viewing the algorithm as a dynamical system, where

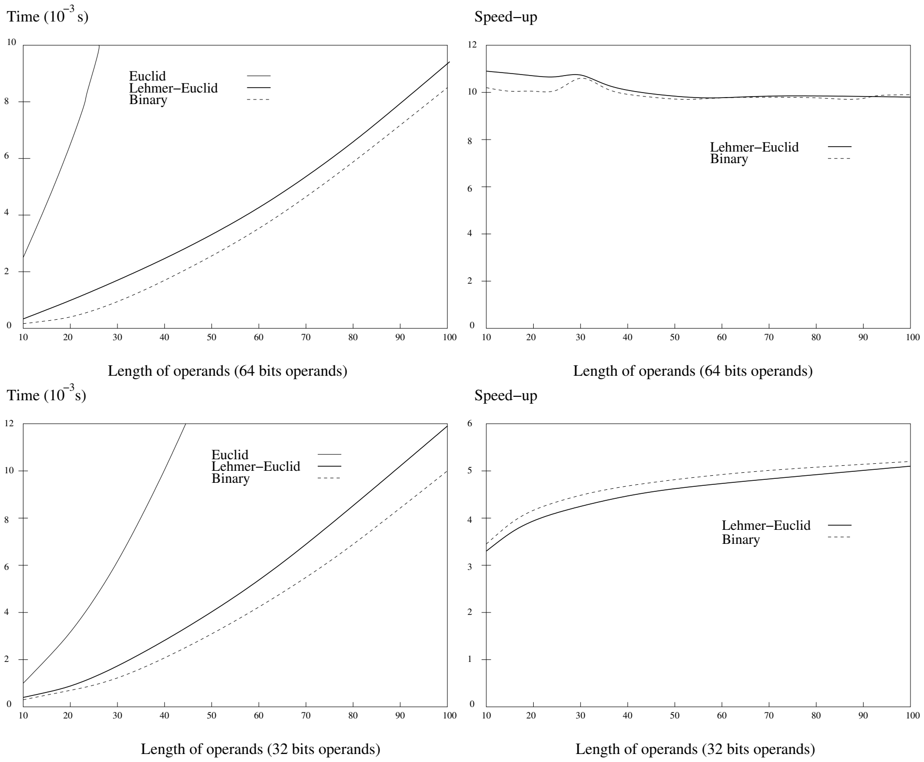

FIGURE 1. Comparison between three algorithms : the plain Euclid Algorithm, the Lehmer-Euclid Algorithm and the Binary Algorithm: Station DEC (top), Station SUN (bottom). Execution time (on the left) and Speed-up (on the right)

each step corresponds to an iteration of the algorithm. More precisely, this method relies on a description of relevant parameters by means of generating functions, a by now common tool in the average-case of algorithms [11]. As is usual in number theory contexts, the generating functions are Dirichlet series. They are first proved to be algebraically related to the so–called transfer operators that encapsulate all the important informations relative to the "dynamics" of the algorithm. The analytical properties of Dirichlet series depend on spectral properties of the transfer operators, most notably the existence of a "spectral gap" that separates the dominant eigenvalue from the remainder of the spectrum. This determines the singularities of the Dirichlet series of costs. The asymptotic extraction of coefficients is then achieved by means of Tauberian theorems [8], so that average complexity estimates finally result. The main thread of the method is thus adequately summarized by the chain:

$$\text{Euclidean algorithm} \rightsquigarrow \text{Associated transformations}$$
$$\rightsquigarrow \text{Transfer operator} \rightsquigarrow \text{Dirichlet series of costs}$$
$$\rightsquigarrow \text{Tauberian inversion} \rightsquigarrow \text{Average-case complexity.}$$

In this way, Vallée studied a whole class of Euclidean algorithms, and this analysis leads to a classification into two subclasses [26] : the first one is formed with slow algorithms of log-squared average complexity, whereas the

other class is formed of fast algorithms, of log average complexity. The same method provided the complete analysis of another widely used algorithm, the Binary algorithm [25]. These methods are also suitable for performing bit-complexity analyses, see [1, 27].
However, all the previous dynamical analyses dealt with algorithms that exhibit a simple structure, so that it is easy to relate the algorithm to the underlying dynamical system. Here, the structure of the Lehmer-Euclid algorithm is more intricate, since the algorithm can be described as a sequence of internal loops. This is why this analysis is also a kind of test for the dynamical analysis methodology.

Plan of the paper. Section 2 is an introductory section that explains the framework, and describes the main algorithms, the Interrupted Euclid Algorithm $\mathcal{E}_\alpha$ with interruption degree α, and the Lehmer–Euclid algorithm $\mathcal{LE}_\mu$ with truncation degree μ. Here, the main cost parameters are defined, and the Theorems are stated. Section 3 describes the general framework of dynamical analysis methodology, and each following section is devoted to the proof of one of the main three theorems: Section 4 for Theorem 1, that describes the average number of iterations of the $\mathcal{E}_\alpha$ algorithm; Section 5 for Theorem 2, that describes the average bit-complexity of the $\mathcal{E}_\alpha$ algorithm; Finally, Section 6 for Theorem 3, that states the main result of this paper, i.e., the average bit-complexity of the $\mathcal{LE}_\mu$ algorithm.

2. The Lehmer-Euclid algorithm.

This introductory Section presents the main algorithms to be studied: the plain Euclidean algorithm, the interrupted Euclidean Algorithm and finally the Lehmer–Euclid Algorithm. We describe here the parameters of interest, and the probabilistic models. We state the three main Theorems.

2.1. The Euclid algorithm.

Let (A_0, A_1) a pair of positive integers with $A_1 \le A_0$. When running on the input (A_0, A_1), the Euclidean algorithm computes the remainder sequence (A_i) defined by

$$A_{i+2} = A_i \mod A_{i+1}, \qquad A_{p+1} = 0,$$

and the last non-zero remainder A_p is the gcd of (A_0, A_1). This computation is then a succession of Euclidean divisions of the form

$$A_i = Q_{i+1}A_{i+1} + A_{i+2} \qquad \text{with} \qquad Q_{i+1} = \lfloor \frac{A_i}{A_{i+1}} \rfloor. \tag{2.1}$$

The integer Q_i is called the i–th quotient and the successive divisions can be written as

$$\mathcal{A}_i = \mathcal{Q}_{i+1}\mathcal{A}_{i+1}, \qquad \text{with} \qquad \mathcal{A}_i := \begin{pmatrix} A_{i+1} \\ A_i \end{pmatrix} \qquad \text{and} \qquad \mathcal{Q}_i := \begin{pmatrix} 0 & 1 \\ 1 & Q_i \end{pmatrix}, \tag{2.2}$$

so that $\mathcal{A}_0 = \mathcal{M}_i\mathcal{A}_i$ with $\mathcal{M}_i := \mathcal{Q}_1\mathcal{Q}_2\cdots\mathcal{Q}_i$.
Additionnally to the quotient sequence (Q_i), the extended Euclidean algorithm computes step by step the matrix $\mathcal{M}_i$ and its inverse $\mathcal{M}_i^{-1}$

$$\mathcal{M}_i^{-1} := \begin{pmatrix} V_{i+1} & U_{i+1} \\ V_i & U_i \end{pmatrix} \qquad \mathcal{M}_i := (-1)^i \begin{pmatrix} U_i & -U_{i+1} \\ -V_i & V_{i+1} \end{pmatrix} \tag{2.3}$$

that involve two other sequences (U_i) and (V_i) defined by

$$\begin{pmatrix} V_1 & U_1 \\ V_0 & U_0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$\text{and} \qquad \begin{pmatrix} V_{i+1} & U_{i+1} \\ V_i & U_i \end{pmatrix} = \begin{pmatrix} -Q_i & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} V_i & U_i \\ V_{i-1} & U_{i-1} \end{pmatrix}.$$

Then, for any $i, 0 \leq i \leq p$, the equality $A_i = A_0 U_i + A_1 V_i$ holds, and the final coefficients U_p and V_p form the Bezout pair (U, V) that satisfies $UA_0 + VA_1 = \gcd(A_0, A_1)$.

2.2. **Main principles of the Lehmer-Euclid algorithm.** In spite of its simplicity, Euclid's algorithm is not well–suited for large integers. Indeed, Euclidean divisions are quite time consuming, especially for large integers. The cost of a division between two large integers may be higher than the cost of the multiplication between two large integers. The idea of Lehmer is the following:

Replace a sequence of large divisions by a sequence of small divisions and small multiplications followed by some large multiplications.

In the sequel, we denote by $\ell(x) := \lfloor \log_2 x \rfloor + 1$ the binary length of a positive integer x. We consider a valid input (A_0, A_1) of the Euclid algorithm. It satisfies $A_1 \leq A_0$, and its length n is (by definition) the binary length of A_0. For some $m \leq n$, the truncated pair (a_0, a_1), defined by

$$a_0 := \lfloor A_0/2^{n-m} \rfloor, \ a_1 := \lfloor A_1/2^{n-m} \rfloor$$

is built by erasing the $n - m$ less significant digits of A_0 and A_1: it is of length m.

In fact, Lehmer suggests to use the first steps of the extended Euclidean algorithm on the small pair (a_0, a_1) to simulate the first steps of the Euclidean algorithm on the large pair (A_0, A_1).

More precisely, the extended Euclidean algorithm is applied to the pair (a_0, a_1), and provides the remainder sequence (a_i), the quotient sequence (q_i) and the two cosequences (u_i) and (v_i). Since the two rationals A_1/A_0 and a_1/a_0 are close, one can expect that the two quotient sequences, the sequence q_i obtained from the Euclidean algorithm on (a_0, a_1) and the sequence Q_i obtained from the Euclidean algorithm on (A_0, A_1) are not too different, at least at the beginning of the process. There always exists some integer $j \geq 0$ for which the two quotient sequences q_i and Q_i are equal for $i \leq j$. Since the equalities $(u_i, v_i) = (U_i, V_i)$ hold for $i \leq j + 1$, it is then possible to compute the large integers A_j and A_{j+1} which would have been obtained when performing j steps of the Euclidean Algorithm on the input (A_0, A_1): The coefficients of matrix $\mathcal{M}_j$ are computed from the small cosequences $(u_i), (v_i)$ so that, with (2.2) and (2.3)

$$A_j = v_j A_0 + u_j A_1, \ A_{j+1} = v_{j+1} A_0 + u_{j+1} A_1.$$

The main problem is now to evaluate a possible value of the index j without computing the quotients Q_i of the large sequence. There exist many different possible tests, due to Lehmer [18], Collins [7] or Jebelean [15], that are in fact very close; we choose the following one:

$$\text{If} \ \ a_j > a_0^{1/2} \qquad \text{then} \qquad Q_i = q_i \qquad \text{for all } i \leq j - 2.$$

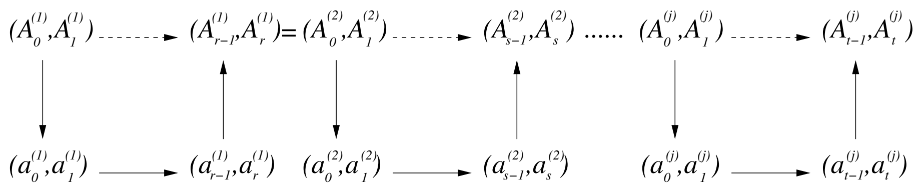

FIGURE 2. General principle of the Lehmer-Euclid algorithm.

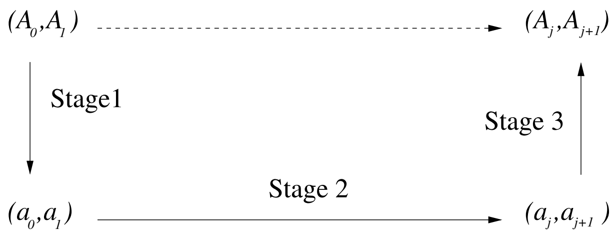

FIGURE 3. Principle of the Lehmer-Euclid algorithm: one phase.

Then, if r is the first index for which $a_r \leq a_0^{1/2}$, then the two sequences Q_i and q_i are the same until $i = r - 3$, and it is thus possible to recover the value of the pair (A_{r-3}, A_{r-2}) without performing the large divisions.

2.3. **The Lehmer-Euclid algorithm.** More precisely, the Lehmer-Euclid algorithm can be viewed as a sequence of phases, each phase replacing a sequence of some steps of the Euclidean Algorithm. Each phase is formed with three different stages (see Figure 3).

(i) **Stage 1.** It is a truncation step which replaces a large pair (A_0, A_1) by the small pair (a_0, a_1).

(ii) **Stage 2.** It can be defined as an "interrupted Euclidean algorithm" that performs only the first iterations of the Euclidean algorithm on input (a_0, a_1). This algorithm stops, after r iterations, as soon as the current integer a_i becomes smaller than $\sqrt{a_0}$.

(iii) **Stage 3.** A multiplication step that recovers the value of the large pair (A_{r-3}, A_{r-2}) which would have been computed with $r - 3$ iterations of the Euclidean Algorithm on the input pair (A_0, A_1).

The Lehmer-Euclid algorithm mainly depends on the choice of the truncation. Here, we work with truncations that are proportional to the length of the input. When the input pair (A_0, A_1) has length n, each truncation step will produce a pair of length m, with $m := \lfloor \mu n \rfloor$ for some fixed parameter $\mu \in]0, 1]$. This parameter μ is called the truncation degree, and the corresponding Lehmer-Euclid algorithm is denoted by $\mathcal{LE}_\mu$.
We work with the set $\widetilde{\Omega}$ of valid inputs and with the subsets $\widetilde{\Omega}_n$ of inputs of length n,

$$\widetilde{\Omega} = \{(A_0, A_1);\ 0 < A_1 \leq A_0, \},$$

$$\widetilde{\Omega}_n := \{(A_0, A_1) \in \widetilde{\Omega};\ \ell(A_0) = n\} = \{(A_0, A_1) \in \widetilde{\Omega};\ 2^{n-1} \leq A_0 < 2^n\}.$$

If the input pair has length n, each truncation step uses the truncation mapping $T_m : \widetilde{\Omega} \to \widetilde{\Omega}_m$ with $m := \lfloor \mu n \rfloor$. This mapping creates, from a

Algorithm $\mathcal{LE}_\mu$.
Input. An integer pair (A, B) with $0 \leq B \leq A$.
Initialisation $n := \ell(A)$; $m := \lfloor \mu n \rfloor$; $A_0 := A$; $A_1 := B$;
While $\ell(A_0) > m$ **do**
  (1). $a_0 := T_m(A_0)$; $a_1 := T_m(A_1)$;
  (2). $i := 1$; $u_0 := 1$; $u_1 := 0$; $v_0 := 0; v_1 := 1$;
    While $a_i > a_0^{1/2}$ **do**
      $q_i := a_{i-1} \operatorname{div} a_i$; $a_{i+1} := a_{i-1} \mod a_i$;
      $u_{i+1} := -u_i q_i + u_{i-1}$; $v_{i+1} := -v_i q_i + v_{i-1}$;
      $i := i + 1$;
  (3). $A' := u_{i-3}A_0 + v_{i-3}A_1$; $B' := u_{i-2}A_0 + v_{i-2}A_1$;
    $A_0 := A'$; $A_1 := B'$;
$i := 1$;
While $A_i > 0$ **do**
  $A_{i+1} := A_{i-1} \mod A_i$;
  $i := i + 1$;
Output. $\gcd(A, B) := A_{i-1}$.

FIGURE 4. The Lehmer-Euclid algorithm with truncation degree μ.

current pair (A', B') of length k, a pair (a', b') of length m obtained by erasing the less $k - m$ significant digits of A' and B',

$$\text{If } k \geq m, \text{ then} \qquad T_m(A', B') := \left(\lfloor A'/2^{k-m} \rfloor, \lfloor B'/2^{k-m} \rfloor\right),$$
$$\text{else} \qquad T_m(A', B') := (0, 0).$$

The Lehmer-Euclid Algorithm $\mathcal{LE}_\mu$ is described in Figure 4.

2.4. **The Interrupted Euclid Algorithm.** We are mainly interested in the analysis of Stage 2, since it provides the value r that determines the length of a phase, i.e., the number of large divisions that can be replaced by small ones. More generally, we wish to study a more general algorithm whose interruption depends on a real parameter $\alpha \in [0, 1]$ (cf Figure 5). When running on an input (a_0, a_1), this algorithm, denoted by $\mathcal{E}_\alpha$, stops as soon as the current integer a_i becomes smaller than a_0^α.

Then, Stage 2 in the Lehmer–Euclid algorithm is exactly the Interrupted Algorithm $\mathcal{E}_{1/2}$ applied on truncated integers. For the moment, we forget the Lehmer–Euclid algorithm and we focus on the analysis of the general Interrupted Algorithm $\mathcal{E}_\alpha$ for any value $\alpha \in [0, 1]$ and any input pair of integers. This algorithm is described in Figure 5.

We work with sets $\Omega, \widetilde{\Omega}$ of valid inputs and with sets $\Omega_n, \widetilde{\Omega}_n$ of valid inputs of binary length n,

$$\Omega = \{(a_0, a_1);\ 0 < a_1 < a_0, \gcd(a_0, a_1) = 1\}, \qquad \widetilde{\Omega} = \{(a_0, a_1);\ 0 < a_1 < a_0\},$$

$$\Omega_n := \{(a_0, a_1) \in \Omega;\ 2^{n-1} \leq a_0 < 2^n\},\ \widetilde{\Omega}_n := \{(a_0, a_1) \in \widetilde{\Omega};\ 2^{n-1} \leq a_0 < 2^n\}.$$

The following definition makes precise the probabilistic model.

Input. An integer pair (a, b) with $b \le a$.
$a_0 := a;\ a_1 := b;$
$u_0 := 1;\ u_1 := 0;\ v_0 := 0;\ v_1 := 1;$
$i := 1;$
While $a_i > a_0^\alpha$ **do**
$\quad q_i := a_{i-1} \operatorname{div} a_i;\ a_{i+1} := a_{i-1} \bmod a_i;$
$\quad u_{i+1} := -u_i q_i + u_{i-1};\ v_{i+1} := -v_i q_i + v_{i-1};$
$\quad i := i + 1$
Output. An integer pair (a_i, a_{i+1}) that satisfies
$a_{i+1} \le a_0^\alpha < a_i$ and $\gcd(a_0, a_1) = \gcd(a_{i+1}, a_i)$.

FIGURE 5. The Interrupted Euclidean Algorithm $\mathcal{E}_\alpha$.

Definition. *Let f be a positive function defined on the unit interval I. We say that Ω (resp. $\widetilde{\Omega}$) is endowed with f if any element (a_0, a_1) of Ω is weighted with the quantity $f(\frac{a_1}{a_0})$.*

This framework defines a sequence of probabilistic models on subsets $\Omega_n, \widetilde{\Omega}_n$. For each n, the corresponding probabilities and expectations on $\Omega_n, \widetilde{\Omega}_n$ are denoted by $\Pr_n, \mathrm{E}_n$; if we wish to insist on the dependence on function f, we put it as an exponent. Then, the symbols $\Pr_n^{(f)}[B]$ (for any subset $B \subset \widetilde{\Omega}$) and $\mathrm{E}_n^{(f)}[X]$ (for any variable X defined on Ω) denote the following quantities

$$\Pr_n^{(f)}[B] := \frac{\sum_{(a_0,a_1)\in\Omega_n\cap B} f(\frac{a_1}{a_0})}{\sum_{(a_0,a_1)\in\Omega_n} f(\frac{a_1}{a_0})}, \mathrm{E}_n^{(f)}[X] := \frac{\sum_{(a_0,a_1)\in\Omega_n} X(a_0,a_1) f(\frac{a_1}{a_0})}{\sum_{(a_0,a_1)\in\Omega_n} f(\frac{a_1}{a_0})}.$$

The exponent (f) will be omitted if necessary.

2.5. **The Interrupted Algorithm. Number of iterations.** Our first main result relates the number P_α of iterations of the algorithm $\mathcal{E}_\alpha$ to the number P of iterations of the Euclidean algorithm $\mathcal{E}$. This Theorem will be proven in Section 4.

Theorem 1. *Suppose that the valid sets $\Omega, \widetilde{\Omega}$ are endowed with some positive function f with bounded variation on the unit interval I. Denote by $\mathcal{E}$ the Euclidean Algorithm, and by $\mathcal{E}_\alpha$ the Interrupted Euclidean Algorithm of parameter α, by P_α the number of iterations of $\mathcal{E}_\alpha$, and by P the number of iterations of $\mathcal{E}$. Then, one has*

(i) For any $\varepsilon > 0$, there exists some $K < 1$ such that

$$\Pr_n\left[\,\left|\frac{P_\alpha}{P} - (1-\alpha)\right| > \varepsilon\right] = O(K^n), \qquad \textit{when } n \to \infty,$$

(ii) The expectations of these costs on Ω_n or on $\widetilde{\Omega}_n$ satisfy, when $n \to \infty$,

$$\mathrm{E}_n[P_\alpha] \sim (1-\alpha)\,\mathrm{E}_n[P] \sim (1-\alpha)\frac{12\log^2 2}{\pi^2} n.$$

Remark. The random variable P was first analyzed by Dixon [9] and Heilbronn [12] around 1970 that proved independently that

$$\mathrm{E}_n[P] \sim \frac{12 \log^2 2}{\pi^2} n,$$

in the case when $f \cong 1$. More recently, always in the particular case when $f \cong 1$, Hensley [13] proved that the random variable P follows asymptotically a normal law. He expressed the expectation and the variance with some function $\Lambda(s)$ which will play a fundamental rôle in this paper,

$$\mathrm{E}_n[P] \sim \frac{-\log 2}{\Lambda'(2)} n, \qquad \mathrm{Var}_n[P] \sim -\log 2 \frac{\Lambda''(2)}{\Lambda'(2)^3} n.$$

Remark. In (i), one can choose

$$K = 2^{-\eta\varepsilon^2} \qquad \text{with} \qquad \eta = |\Lambda'(2)|/\Lambda''(2).$$

2.6. **The Interrupted Euclidean Algorithm. Bit-complexity.** The operations that are performed by the Euclidean algorithm $\mathcal{E}$ have not all the same cost. The Extended Euclidean algorithm performs exchanges, multiplications and divisions. We denote by M the constant that arises in the mutiplication and exchange costs, and by D the constant that arises in the division cost. Then the cost of a multiplication between two integers u and v has a cost equal to $M \cdot \ell(u) \cdot \ell(v)$ while the cost of an exchange between u and v is $M(\ell(u) + \ell(v))$.
A Euclidean division $v = uq + r$ has a bit-cost equal to $D\ell(u) \cdot \ell(q)$. It is followed with two exchanges, whose cost is $2M\ell(u)$. When the algorithm $\mathcal{E}$ performs p iterations on input (a_0, a_1), the total bit cost B of the execution of the algorithm

$$B(a_0, a_1) = \sum_{i=1}^{p} \ell(a_i) \cdot b(q_i) \qquad \text{with} \qquad b(q) := D\ell(q) + 2M \tag{2.4}$$

both involves the quotient sequence (q_i) and the remainder sequence (a_i).
For computing the Bezout coefficients, the Extended Euclidean algorithm performs multiplications and exchanges, so that the supplementary bit-cost C for one Bezout coefficient v_i

$$C(a_0, a_1) = \sum_{i=1}^{p} \ell(v_i) \cdot c(q_i) \qquad \text{with} \qquad c(q) := (\ell(q) + 2)M \tag{2.5}$$

both involves the quotient sequence (q_i) and the cosequence (v_i).

Theorem 1 proves that the algorithm $\mathcal{E}_\alpha$ stops (almost surely) at the $\lfloor (1-\alpha)P \rfloor$-th iteration. We introduce another truncated algorithm, that we denote by $\overline{\mathcal{E}}_\alpha$ that exactly stops at the $\lfloor (1-\alpha)P \rfloor$-th iteration: It is a regularized version of the $\mathcal{E}_\alpha$ algorithm. This algorithm –not very realistic, since it must "guess" the value of P– is just a tool for the analysis; The following Theorem shows that it behaves asymptotically in the same way as $\mathcal{E}_\alpha$ and it is easier to analyse it. This Theorem will be proven in Section 5.

Theorem 2. *Suppose that the valid sets $\Omega, \widetilde{\Omega}$ are endowed with some function f with bounded variation on the unit interval I. Denote by $\mathcal{E}$ the*

Euclidean Algorithm, and by $\mathcal{E}_\alpha$ the Interrupted Euclidean Algorithm of parameter α. We consider three measures of cost for the Interrupted Algorithm $\mathcal{E}_\alpha$: the bit-cost B_α, the supplementary bit-cost C_α due to the computation of one Bezout coefficient, and the total bit-cost E_α of the extended Euclidean algorithm (i.e., $E_\alpha = B_\alpha + 2C_\alpha$). The same quantities with a bar denote the corresponding costs for the $\overline{\mathcal{E}}_\alpha$ algorithm, and the quantities B, C, E denote the same quantities for the Euclidean algorithm $\mathcal{E}$.
Then, for any positive function f with bounded variation on I, one has, for $n \to \infty$,

$$\begin{array}{llll} (i) & \mathrm{E}_n[B_\alpha] & \sim \ \mathrm{E}_n[\overline{B}_\alpha] \sim & (1-\alpha^2)\,\mathrm{E}_n[B] \\ (ii) & \mathrm{E}_n[C_\alpha] & \sim \ \mathrm{E}_n[\overline{C}_\alpha] \sim & (1-\alpha)^2\,\mathrm{E}_n[C] \\ (iii) & \mathrm{E}_n[E_\alpha] & \sim \ \mathrm{E}_n[\overline{E}_\alpha] \sim & \frac{1}{3}(1-\alpha)(3-\alpha)\,\mathrm{E}_n[E] \end{array}$$

Remark. The random variables B, C, E were first analyzed by Akhavi and Vallée [1], [27] in the case when $f \cong 1$. With the notations used here, their results can be translated in the following way:

$$\mathrm{E}_n[B] \sim (L_1 M + L_2 D)\, n^2, \qquad \mathrm{E}_n[C] \sim (L_1 + L_2) M\, n^2$$

$$\text{with} \qquad L_1 = \frac{12 \log^2 2}{\pi^2} \sim 0.58, \qquad L_2 = \frac{6 \log 2}{\pi^2} \log \prod_{k=0}^{\infty} (1 + \frac{1}{2^k}) \sim 0.66.$$

In the sequel of this Section, we come back to the Lehmer-Euclid Algorithm $\mathcal{LE}_\mu$. We recall that we work with a truncation degree μ: From an input (A_0, A_1) of $\widetilde{\Omega}_n$, we deal with pairs of length m with $m := \lfloor \mu n \rfloor$, and we always use the truncation map T_m that creates a new input (a_0, a_1) that belongs to $\widetilde{\Omega}_m$. Here, we wish to give an idea of what we can expect about the asymptotic behaviour of the Lehmer-Euclid Algorithm, if we "insert" inside it our previous results about the Interrupted Algorithm $\mathcal{E}_\alpha$. And the sequel of the paper –namely Section 6– will prove these facts.

2.7. **Comparison between the Lehmer-Euclid Algorithm and the Euclid Algorithm. The first phase.** If the algorithm starts with $f \cong 1$, the distribution is uniform for the inputs (A_0, A_1) in $\widetilde{\Omega}_n$, and the distribution of the truncated pairs (a_0, a_1) is also uniform on $\widetilde{\Omega}_m$.
Then Stage 2 is exactly the Interrupted algorithm $\mathcal{E}_{1/2}$ on integers of length $m := \lfloor \mu n \rfloor$, and stops when the length of the current integers a_i, u_i, v_i is about $m/2$. In Stage 3, we compute four products, each between an integer of length n and an integer of length $m/2$. (see Figure 3). The output integers (A', B') have now a length about $n(1 - \frac{\mu}{2})$. Consequently, the top horizontal line is the Interrupted Algorithm $\mathcal{E}_{1-(\mu/2)}$ on integers of length n.
Then, we can easily compare the two costs, and the expressions involve the length n of the inputs, the degree μ of integer truncation, the constant M that arises in a multiplication cost, and the constant D that arises in a division cost. The constants L_1, L_2 are the constants that intervene in the Euclidean bit–cost.

The cost of the top horizontal line corresponding to the first phase is equal to:

$$\left(1-(1-\frac{\mu}{2})^2\right)(L_1M+L_2D)n^2=\left[(\mu-\frac{\mu^2}{4})(L_1M+L_2D)\right]n^2.$$

This cost has to be compared with the total cost of the first phase (Stages 2 and 3), that is equal to

$$\left[\frac{3}{4}(L_1M+L_2D)\mu^2+\frac{1}{2}(L_1+L_2)M\mu^2+2\mu M\right]n^2.$$

2.8. **Comparison between the Lehmer-Euclid Algorithm and the Euclid Algorithm. The other phases.** During each phase, the length of the small pair decreases of a quantity equal to $m/2$. If it is also true for the large pair, the average number of phases will be around $2/\mu$. At the end of the j-th phase, one computes four products, each between an integer of length $m/2$ and an integer of length $(1-(j-1)(\mu/2))n$ (that is the length of the large integers at the beginning of the j-th phase). If the argument given in the previous paragraph could be repeated for each phase, the average cost of the j-th phase would be

$$\frac{3}{4}(L_1M+L_2D)\mu^2n^2+\frac{1}{2}(L_1+L_2)M\mu^2n^2+4\frac{\mu}{2}(1-(j-1)\frac{\mu}{2})Mn^2,$$

and the average cost of the Lehmer-Euclid algorithm would be

$$\left[\frac{3}{2}(L_1M+L_2D)\mu+(L_1+L_2)M\mu+(2-\mu)M\right]n^2,$$

to be compared to the bit–cost of the classical Euclidean Algorithm, that is $(L_1M+L_2D)n^2$.

2.9. **The final result.** However, the previous argument cannot be repeated a priori for the other phases, because the distribution on Ω is modified by the execution of the Euclidean algorithm. At the beginning of the second phase, the distribution of the new inputs at Stage 1 is not the same as at the beginning of the algorithm. However, the evolution of the distribution of the integers during the execution of the Euclidean Algorithm can be precisely described with tools of Dynamical Analysis. The main idea is then to simulate the cost of the Lehmer-Euclid algorithm only on the Euclid algorithm itself. Then all the previous remarks will be proven and we shall obtain our final result:

Theorem 3. *Suppose that the valid sets $\Omega, \widetilde{\Omega}$ are endowed with some function f with bounded variation on the unit interval I. When the Lehmer-Euclid algorithm deals with a truncation degree μ, its average bit-complexity on pairs of length n is asymptotically equal to*

$$\left[\frac{3}{2}(L_1M+L_2D)\mu+(L_1+L_2)M\mu+(2-\mu)M\right]n^2.$$

Here, L_1 and L_2 are the two constants that appear in the average bit complexity of the Euclidean algorithm,

$$L_1=\frac{12\log^2 2}{\pi^2}\sim 0.58, \qquad L_2=\frac{6\log 2}{\pi^2}\log\prod_{k=0}^{\infty}(1+\frac{1}{2^k})\sim 0.66,$$

and M, D the constants that intervene in a cost of a multiplication or a division.

In the sequel of the paper, we shall prove successively the three Theorems. We first recall the general methodology of what we call a dynamical analysis.

3. Principles of dynamical analysis.

This method uses tools that come from dynamical systems theory, mainly transfer operators.

3.1. **Dirichlet series and Tauberian Theorem.** We are interested in analysing some costs, and, in the sequel, we deal with the generating Dirichlet series of these costs. To any cost X, and to any weight defined on I, $\widetilde{\Omega}$, we associate Dirichlet series

$$F_X(s) = \sum_{(a_0,a_1)\in\Omega} \frac{X(a_0,a_1)}{a_0^s} f(\frac{a_1}{a_0}), \qquad \widetilde{F}_X(s) = \sum_{(a_0,a_1)\in\widetilde{\Omega}} \frac{X(a_0,a_1)}{a_0^s} f(\frac{a_1}{a_0}).$$

Then,

$$F_X(s) = \sum_{a\geq 1} \frac{x_a}{a^s}, \qquad \widetilde{F}_X(s) = \sum_{a\geq 1} \frac{\widetilde{x}_a}{a^s},$$

where x_a, $\widetilde{x}_a$ denote the cumulative costs of X on

$$\omega_a := \{(a_0,a_1)\in\Omega, a_0 = a\}, \qquad \widetilde{\omega}_a := \{(a_0,a_1)\in\widetilde{\Omega}, a_0 = a\}.$$

For the trivial cost, t_a or $\widetilde{t}_a$ are just the weights of subsets $\omega_a, \widetilde{\omega}_a$. The mean values of the cost X on $\Omega_n, \widetilde{\Omega}_n$ are then given by the ratio of partial sums,

$$\mathrm{E}_n[X] = \frac{\sum_{\ell(a)=n} x_a}{\sum_{\ell(a)=n} t_a}, \qquad \widetilde{\mathrm{E}}_n[X] = \frac{\sum_{\ell(a)=n} \widetilde{x}_a}{\sum_{\ell(a)=n} \widetilde{t}_a} \tag{3.1}$$

The asymptotics of such partial sums can be provided when applying the following Tauberian theorem, due to Delange [8].

Tauberian Theorem. [Delange] *Let $F(s)$ be a Dirichlet series with non negative coefficients*

$$F(s) = \sum_{a\geq 1} \frac{x_a}{a^s}.$$

Assume that

(i) $F(s)$ converges for $\Re(s) > \sigma > 0$ and is analytic on $\Re(s) = \sigma, s \neq \sigma$,

(ii) for some $\theta \geq 0$, one has, for s near σ,

$$F(s) = \frac{A(s)}{(s-\sigma)^{\theta+1}} + C(s),$$

where A, C are analytic at σ, with $A(\sigma) \neq 0$.
Then, as $n \to \infty$,

$$\sum_{\ell(a)=n} x_a = \left[(1-\frac{1}{2^\sigma})(\log 2)^\theta \frac{A(\sigma)}{\sigma\Gamma(\theta+1)}\right] 2^{n\sigma} n^\theta \quad [1+\varepsilon(n)], \quad \varepsilon(n)\to 0.$$

3.2. **Costs of interest.** We now describe the main costs that intervene in this paper. Theorem 1 deals with the parameter P_α that denotes the number of iterations of the $\mathcal{E}_\alpha$ Algorithm, and we are mainly interested by the events $[P_\alpha > i]$ and $[P_\alpha \le i]$. Since the algorithm stops as soon as $a_i \le a_0^\alpha$, these events satisfy

$$[P_\alpha > i] = \left[\frac{a_i}{a_0^\alpha} > 1\right], \qquad [P_\alpha \le i] = \left[\frac{a_i}{a_0^\alpha} \le 1\right],$$

and Markov's inequality entails

$$\mathrm{Pr}_n\left[P_\alpha > i\right] \le \mathrm{E}_n\left[\left(\frac{a_i}{a_0^\alpha}\right)^\gamma\right], \ \forall \gamma > 0, \tag{3.2}$$

$$\mathrm{Pr}_n\left[P_\alpha \le i\right] \le E_n\left[\left(\frac{a_i}{a_0^\alpha}\right)^\gamma\right], \ \forall \gamma < 0. \tag{3.3}$$

We are then led to define the first cost as

$$M(a_0, a_1) = \left(\frac{a_i}{a_0^\alpha}\right)^\gamma. \tag{3.4}$$

Section 2.6 explained why the main costs to be studied in Theorem 2, namely $\overline{B}_\alpha$ and $\overline{C}_\alpha$, involve the main parameters of the Euclidean algorithm: the depth P, and the four main sequences that are computed, the quotients (q_i), the remainders (a_i), and the two cosequences(u_i), (v_i), under the form

(3.5)
$$\overline{B}_\alpha(a_0, a_1) \sim \sum_{i=1}^{\lfloor(1-\alpha)p\rfloor} \ell(a_i)\cdot b(q_i), \qquad \overline{C}_\alpha(a_0, a_1) \sim \sum_{i=1}^{\lfloor(1-\alpha)p\rfloor} \ell(u_i)\cdot c(q_i)$$

with

$$b(q) := \ell(q)D + 2M, \qquad c(q) := (\ell(q)+2)M, \tag{3.6}$$

We shall provide alternative expressions for the corresponding Dirichlet series of costs in Sections 4 and 5. These expressions deal with the transfer operator $\mathbf{H}_s$ relative to the Euclidean dynamical system. Then, the singularities of the Dirichlet series will become apparent and related to the dominant spectral objects of the transfer operator $\mathbf{H}_s$. We now describe this dynamical system and present the operator $\mathbf{H}_s$ together with its main spectral properties.

3.3. **The Euclidean Dynamical system.** When computing the gcd of the integer-pair (a_0, a_1), Euclid's algorithm performs a sequence of p iterations of the form

$$a_0 = q_1 a_1 + a_2, \ \ a_1 = q_2 a_2 + a_3, \ldots, \ \ a_{p-1} = q_p a_p. \tag{3.7}$$

This sequence can be associated with the map S that satisfies

$$S\left(\frac{a_1}{a_0}\right) = \frac{a_2}{a_1}, \ldots, S\left(\frac{a_i}{a_{i-1}}\right) = \frac{a_{i+1}}{a_i}, \ldots, S\left(\frac{a_p}{a_{p-1}}\right) = 0.$$

When extended to the real interval $I = [0,1]$, this map corresponds to the classical continued fraction expansion algorithm, and is defined by

$$S(x) = \frac{1}{x} - \lfloor\frac{1}{x}\rfloor, \qquad \text{or by } S(x) = \frac{1}{x} - q \text{ for } x \in [\frac{1}{q+1}, \frac{1}{q}].$$

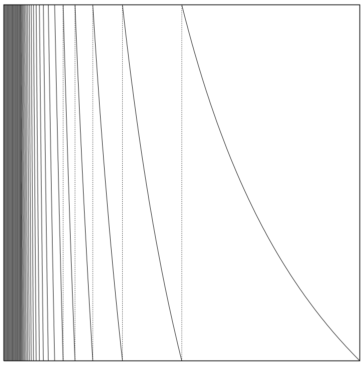

FIGURE 6. Euclidean dynamical system

The pair (I, S) defines the dynamical system relative to Euclid algorithm. We denote by $\mathcal{H}$ the set of the inverse branches of S,

$$\mathcal{H} = \{x \to \frac{1}{q+x}; q \geq 1\},$$

and by $\mathcal{H}^n$ the set of inverse branches of depth n (i.e., the set of inverse branches of S^n),

$$\mathcal{H}^n = \{h = h_1 \circ \cdots \circ h_n; h_i \in \mathcal{H}, \forall i\}.$$

Then, the sequence (3.7) builds a continued fraction

$$\frac{a_1}{a_0} = \cfrac{1}{q_1 + \cfrac{1}{q_2 + \cfrac{1}{q_3 + \cfrac{1}{\ddots + \cfrac{1}{q_p}}}}}, \tag{3.8}$$

and can be written as

$$\frac{a_1}{a_0} = h(0) \qquad \text{with} \qquad h = h_1 \circ h_2 \circ \ldots \circ h_p \in \mathcal{H}^p. \tag{3.9}$$

One then associates to each execution of the algorithm a unique LFT h whose depth is exactly the number p of divisions performed.
Remark that all quantities of interest can be read on the continued fraction (3.8) or in the decomposition (3.9). The depth P equals its height; remark that the i-th LFT h_i used by the algorithm is exactly the LFT relative to matrix $\mathcal{Q}_i$ of Section 2.1, so that the LFT $h_1 \circ h_2 \circ \ldots \circ h_i$ is relative to matrix $\mathcal{M}_i$ of Section 2.1. Then, when one "splits" the CF-expansion (3.8)

of a_1/a_0 at depth i, one obtains two CF-expansions

(3.10)
$$\cfrac{1}{q_1 + \cfrac{1}{q_2 + \cfrac{1}{q_3 + \cfrac{1}{\ddots + \cfrac{1}{q_{i-1}}}}}}, \qquad \cfrac{1}{q_i + \cfrac{1}{q_{i+1} + \cfrac{1}{q_{i+2} + \cfrac{1}{\ddots + \cfrac{1}{q_p}}}}},$$

defining each a rational number: When one considers only the $i-1$ first LFT's, one gets the "beginning" rational that can be expressed with the co-sequences $(u_i), (v_i)$ as

$$h_1 \circ h_2 \circ \cdots \circ h_{i-1}(0) = \frac{|u_i|}{|v_i|}.$$

When one considers only the $p-i$ last LFT's, one gets the "ending" rational that can be expressed with the sequence (a_i),

$$h_{i+1} \circ h_{i+2} \circ \cdots \circ h_p(0) = \frac{a_{i+1}}{a_i}.$$

3.4. **Transfer operators.** The main tool of dynamical analysis is the transfer operator, denoted by $\mathbf{H}_s$. It generalizes the density transformer $\mathbf{H}$ that describes the evolution of the density: if $f = f_0$ denotes the initial density on I, and f_1 the density on $\mathcal{I}$ after one iteration of S, then f_1 can be written as $f_1 = \mathbf{H}[f_0]$, where $\mathbf{H}$ is defined by

$$\mathbf{H}[f](x) = \sum_{h \in \mathcal{H}} |h'(x)| \, f \circ h(x). \tag{3.11}$$

Here, the "component" operator $\mathbf{R}_h$ denotes the operator relative to a single LFT $h \in \mathcal{H}$

$$\mathbf{R}_h[f](x) = |h'(x)| \, f \circ h(x). \tag{3.12}$$

The derivative $h'(x)$ can be expressed with the denominator function D defined by

$$D[g](x) = cx + d, \ \text{for } g(x) = \frac{ax+b}{cx+d} \ \text{with} \quad \gcd(a,b,c,d) = 1,$$

as

$$h'(x) = \frac{\det h}{D[h](x)^2}.$$

Since all the LFT's have a determinant equal to ± 1, this entails an alternative expression for $\mathbf{R}_h$,

$$\mathbf{R}_h[f](x) = \sum_{h \in \mathcal{H}} \frac{1}{D[h](x)^2} \, f \circ h(x).$$

It is convenient to use a more general operator that depends on a complex parameter s

$$\mathbf{R}_{s,h}[f](x) := \frac{1}{D[h](x)^s} \, f \circ h(x).$$

The composition property of the denominator,

$$D[h \circ g](x) = D[h](g(x)) D[g](x)$$

entails a composition property on operators, namely

$$\mathbf{R}_{s,h\circ g} = \mathbf{R}_{s,g} \circ \mathbf{R}_{s,h}.$$

In particular, the transfer operator $\mathbf{H}_s$, and its iterates are defined as

$$\mathbf{H}_s = \sum_{h\in\mathcal{H}} \mathbf{R}_{s,h}, \qquad \mathbf{H}_s^n = \sum_{h\in\mathcal{H}^n} \mathbf{R}_{s,h}.$$

Since $\mathbf{H}_2 = \mathbf{H}$, the transfer operator can be viewed as a generalization of the density transformer. Finally, the operator that is the quasi-inverse of $\mathbf{H}_s$,

$$(\mathbf{I} - \mathbf{H}_s)^{-1} := \sum_{n\geq 0} \mathbf{H}_s^n = \sum_{h\in\mathcal{H}^\star} \mathbf{R}_{s,h}$$

will play a fundamental rôle in the sequel, and it can be viewed as a generating operator of the set $\mathcal{H}^\star$ formed by all possible transformations used by the Euclidean Algorithm.

3.5. **Spectral properties of the transfer operator.** In dynamical analysis context, singularity analysis of generating functions is closely related to spectral properties of the transfer operator. These properties depend on the functional space where the operator acts. Previous analyses of Euclidean algorithms dealt with spaces of analytic functions defined on a complex neighborhood of the unit interval I. Here, we have to work with characteristic functions of some intervals, and we are led to work with a larger space, the space of functions with bounded variation on the unit interval I. This functional space was already previously used in dynamical analysis [5], and the main properties of the transfer operator, when acting on this functional space, can be found there.

For $\Re(s) > 1$, the operator $\mathbf{H}_s$ acts on $BV(I)$ and the map $s \to \mathbf{H}_s$ is analytic. We denote by $R(s)$ the spectral radius of $\mathbf{H}_s$. The function $s \to R(s)$ is strictly decreasing along the real axis, and satisfies $R(s) \leq R(r)$ for $\Re(s) = r$. For $s = 2$, the operator is quasi–compact: there exists a spectral gap between the unique dominant eigenvalue (that equals 1, since the operator is a density transformer) and the remainder of the spectrum. By perturbation theory, these facts —existence of a dominant eigenvalue $\lambda(s)$ and of a spectral gap— remain true in a neighborhood of $s = 2$. There, the operator splits into two parts: the part relative to the dominant eigensubspace, denoted $\mathbf{P}_s$, and the part relative to the remainder of the spectrum, denoted $\mathbf{N}_s$, whose spectral radius is strictly less than $|\lambda(s)|$. This leads to the following spectral decomposition

$$\mathbf{H}_s[f](x) = \lambda(s)\mathbf{P}_s[f](x) + \mathbf{N}_s[f](x). \tag{3.13}$$

The projector $\mathbf{P}_s$ can also be written as $\mathbf{P}_s[f](x) = \psi_s(x)E_s[f]$ where ψ_s is the dominant eigenfunction and E_s is the dominant eigenfunction of the dual operator $\mathbf{H}_s^\star$ with normalization condition $E_s[\psi_s] = 1$.
The decomposition (3.13) extends to the powers of the operator

$$\mathbf{H}_s^n[f](x) = \lambda^n(s)\mathbf{P}_s[f](x) + \mathbf{N}_s^n[f](x),$$

and finally to the quasi-inverse $(\mathbf{I} - \mathbf{H}_s)^{-1}$

$$(\mathbf{I} - \mathbf{H}_s)^{-1}[f](x) = \frac{\lambda(s)}{1-\lambda(s)}\mathbf{P}_s[f](x) + (\mathbf{I} - \mathbf{N}_s)^{-1}[f](x). \tag{3.14}$$

The first term on the right is singular at $s = 2$, while the second term is analytic on the half–plane $\{\Re(s) > 2\}$.

3.6. **Decomposition of the quasi inverse. Properties of the dominant eigenvalue.** We summarize here the main properties that will be extensively used in the sequel, in particular, when one applies the Tauberian Theorem.

When the operator $\mathbf{H}_s$ acts on $BV(I)$, the following is true:
(a) At $s = 2$, dominant spectral objects are all explicit,

$$\lambda(2) = 1, \qquad \psi_2(x) = \frac{1}{\log 2}\frac{1}{1+x},$$

$$E_2[f] = \int_0^1 f(t)dt, \qquad \lambda'(2) = \frac{-\pi^2}{12\log 2}.$$

(b) The dominant eigenvalue $s \to \lambda(s)$ is well–defined on a neighborhood $\mathcal{V}$ of $s = 2$. On the interval $\mathcal{V} \cap \mathbf{R}$, the function $s \to \lambda(s)$ is positive, analytic, strictly decreasing and strictly log-*convex.*
(c) For any $\sigma > 1$ and any s with $\Re(s) = \sigma$, the spectral radius $R(s)$ satisfies $R(s) \le R(\sigma)$. For any $\sigma \in \mathcal{V} \cap \mathbf{R}$ and any s with $\Re(s) = \sigma, s \ne \sigma$, the spectral radius $R(s)$ satisfies $R(s) < R(\sigma) = \lambda(\sigma)$.
(d) The quasi–inverse $(I - \mathbf{H}_s)^{-1}$ of $\mathbf{H}_s$ is analytic on the half–plane $\{\Re(s) \ge 2, s \ne 2\}$, and has a simple pole at $s = 2$. Near $s = 2$, one has, for any positive function f of $BV(I)$, and any $x \in I$,

$$(I - \mathbf{H}_s)^{-1}[f](x) \sim \frac{1}{s-2}\,\frac{-1}{\lambda'(2)}\,\frac{1}{\log 2}\,\frac{1}{1+x}\left(\int_I f(t)dt\right). \tag{3.15}$$

3.7. **The Basic Dirichlet series $F_1(s)$.** We explain now, as a kind of test, how the previous results can be used in the study of the Dirichlet series $F_1(s)$ relative to the trivial cost $X = 1$. First, this series admits an other expression that involves the quasi-inverse $(I - \mathbf{H}_s)^{-1}$. Consider an input (a_0, a_1) of Ω. There exists a unique LFT h of $\mathcal{H}^\star$ for which $a_1/a_0 = h(0)$. Then

$$F_1(s) = \sum_{(a_0,a_1)\in\Omega} \frac{1}{a_0^s} f(\frac{a_1}{a_0}) = \sum_{h\in\mathcal{H}^\star} \frac{1}{D[h](0)^s} f\circ h(0) = (I - \mathbf{H}_s)^{-1}[f](0).$$

Moreover, the Riemann Zeta function $\zeta(s)$ relates $F_1(s)$ and $\widetilde{F}_1(s)$ via the equality $\widetilde{F}_1(s) = \zeta(s)F_1(s)$. Then, the Tauberian Theorem applies to $F_1(s), \widetilde{F}_1(s)$ with $\sigma = 2, \theta = 0$, and, in the case when f is a density, one obtains

$$F_1(s) \sim \frac{1}{(s-2)}\,\frac{-1}{\lambda'(2)}\,\frac{1}{\log 2} = \frac{1}{(s-2)}\,\frac{6}{\pi^2}$$

so that

$$\sum_{\ell(a)=n} t_a \sim \frac{9}{2\pi^2}4^n, \qquad \sum_{\ell(a)=n} \widetilde{t}_a \sim \frac{3}{4}4^n.$$

Of course, these results can be obtained directly! But, the previous lines are in a sense "generic" in the dynamical analysis methods.

3.8. **Evolution of the density during the execution of the Euclidean Algorithm.** When all the real inputs of the Continued Fraction Algorithm are considered, the evolution of the density during the algorithm is well-known. In this case, the algorithm does not terminate (almost surely), and the asymptotic density on the unit interval is the Gauss density $\psi = \psi_2$ defined in 3.6,

$$\psi(t) := \frac{1}{\log 2}\frac{1}{1+t}.$$

But, when one only considers rationals of Ω, the situation is not so clear, since the algorithm always terminates; moreover, at the end of the algorithm, all the rationals are now at the point 0, so the limit measure on Ω is the Dirac measure concentrated at $x = 0$.

The following two lemmas describe the evolution of density on rational inputs; the first one at the beginning of the algorithm, and the second one at any fraction of the depth.

Lemma 1. *Suppose that Ω is endowed with a density f with bounded variation. Then, for any interval B, one has*

$$\lim_{n\to\infty} \Pr_n[B] = \int_B f(t)dt.$$

Proof. Since Ω is endowed with f, one has:

$$\Pr_n[B] = \frac{\sum_{x\in\Omega_n\cap B} f(x)}{\sum_{x\in\Omega_n} f(x)} = \frac{\sum_{x\in\Omega_n} \mathbf{1}_B(x)f(x)}{\sum_{x\in\Omega_n} f(x)}.$$

The Dirichlet series of costs for numerator and denominator are

$$G(s) = \sum_{(a_0,a_1)\in\Omega} \frac{1}{a_0^s}(\mathbf{1}_B f)(\frac{a_1}{a_0}), \qquad F_1(s) = \sum_{(a_0,a_1)\in\Omega} \frac{1}{a_0^s} f(\frac{a_1}{a_0}),$$

that have alternative expressions that involve the quasi-inverse $(I - \mathbf{H}_s)^{-1}$,

$$F_1(s) = (I - \mathbf{H}_s)^{-1}[f](0), \qquad G(s) = (I - \mathbf{H}_s)^{-1}[\mathbf{1}_B f](0)$$

Then, Tauberian Theorem applies to $F(s)$ and $G(s)$ with $\sigma = 2$ and $\theta = 0$ and gives the result. ∎

Now, the next lemma describes the situation "at a fraction of the depth":

Lemma 2. *Suppose that Ω is endowed with a density f with bounded variation. For $\alpha \in]0,1[$ and any $x \in \Omega$ with depth p, denote by x_α the pair that arises at the $\lfloor \alpha p \rfloor$–th iteration of the Euclidean algorithm when applied to x. Then, for any interval B,*

$$\lim_{n\to\infty} \Pr_n[x_\alpha \in B] = \int_B \psi(t)dt \qquad \textit{where} \qquad \psi(x) = \frac{1}{\log 2}\frac{1}{1+x}$$

is the stationary density of the Euclidean Algorithm.

Proof. Consider a rational $x := a_1/a_0$ of depth p and its transform $g(x)$ after $i := i(p)$ steps of the Euclidean algorithm. Then $g(x) = a_{i+1}/a_i$. There exists some LFT's h of depth i, r of depth $p-i$ for which $x := h \circ r(0), g(x) := r(0)$. Then

$$\Pr_n[x; g(x) \in B] = \frac{\sum_{x\in\Omega_n} \mathbf{1}_B \circ g(x)f(x)}{\sum_{x\in\Omega_N} f(x)}.$$

α is related to the truncation in the Euclidean Algorithm; $\beta := 1 - \alpha$.
γ is strictly positive; it appears when using Markov's inequality.
s denotes the "Dirichlet series parameter". For a fixed value of the pair (α, γ), we let $s^- = s - \beta\gamma$ and $s^+ = s + \alpha\gamma$.
P denotes the number of iterations of the Euclidean Algorithm, (the depth).
The integer i is the index of a generic step in the Euclidean Algorithm, $0 \le i \le P$, of the form $i = \lfloor \delta P \rfloor$, (with $\delta \in [0,1]$).
For easier computations, δ is a rational $\delta = c/(c+d)$. Thus, $P = (c+d)k + j$, with $j < c + d$, and one sets $j' := \lfloor \delta j \rfloor$,

FIGURE 7. The notations used in the Proof of Theorem 1.

Since $\mathbf{1}_B \circ g(x) f(x) = \mathbf{1}_B \circ r(0) f \circ h \circ r(0)$, the Dirichlet series relative to the numerator is

$$G(s) = \sum_p \mathbf{H}_s^{p-i(p)} \left[\mathbf{1}_B \mathbf{H}_s^{i(p)}[f] \right](0).$$

It converges on the punctured half–plane $\{\Re(s) > 2, s \neq 2\}$. Moreover, in the case when $i(p)$ and $p - i(p)$ both tend to ∞ for $p \to \infty$, the "dominant" part of this Dirichlet series at $s = 2$ is

$$G^+(s) = \sum_p \lambda(s)^p \mathbf{P}_s \left[\mathbf{1}_B \mathbf{P}_s[f] \right](0);$$

it has a pole at $s = 2$, whose residue is

$$\frac{-1}{\lambda'(2)} \mathbf{P}_2 \left[\mathbf{1}_B \mathbf{P}_2[f] \right](0) = \frac{-1}{\lambda'(2)} \psi(0) \int_I \mathbf{1}_B(t) \mathbf{P}_2[f](t) dt.$$

Since $\mathbf{P}_2[f](t) = \psi(t) dt$, one finally obtains the result. ■

We thus can describe (in an approximate way) the distribution at the beginning of each phase of the Lehmer–Euclid algorithm. This provides a first argument towards our heuristic reasoning for Section 2.8.

Proposition 1. *Suppose that the set of the initial inputs of the Euclidean Algorithm is endowed with some density f with bounded variation on the unit interval I. Then, at the beginning of each phase of the Lehmer-Euclid algorithm –except the first one–, the large inputs and the small inputs have a distribution that is "close" to the distribution of Ω weighted by the stationary density of the Euclidean Algorithm.*

4. Analysis of the Interrupted Euclidean Algorithm. Number of Iterations.

In this Section, we prove Theorem 1. We deal here with the cost M defined in (3.4) and we denote by $F_M, \widetilde{F}_M$ the Dirichlet series relative to this cost. We use in this Section the notations that are gathered in Figure 7.

4.1. **The Dirichlet series $F_M(s)$.** We first provide an expression for $F_M(s)$

$$F_M(s) := \sum_{(a_0,a_1)\in\Omega} \frac{a_i^\gamma}{a_0^{\alpha\gamma+s}}$$

as a function of operator $\mathbf{H}_t$ that involves two values of t, namely $t = s^+$ and $t = s^-$.

Lemma 3. *The Dirichlet series $F_M(s), \widetilde{F}_M(s)$ can be expressed in terms of transfer operator,*

$$F_M(s) = \sum_{p\geq i} \left(\mathbf{H}_{s^-}^{p-i} \circ \mathbf{H}_{s^+}^{i}[f](0)\right), \qquad \widetilde{F}_M(s) = \zeta(s^+)F_M(s). \tag{4.1}$$

Proof. Consider an input (a_0, a_1) of Ω on which the algorithm performs p iterations. There exists a unique LFT h of depth p such that $a_1/a_0 = h(0)$. One can decompose h in two LFT's g and r of depth i and $p-i$ such that $h = g\circ r$. In this case, one has $D[g\circ r](0) = a_0$ and $D[r](0) = a_i$. Now, the general term of the series $F_M(s)$ decomposes as

$$\frac{a_i^\gamma}{a_0^{\alpha\gamma+s}} = \frac{D[r]^\gamma(0)}{D[g\circ r]^{s^+}(0)} = \frac{1}{D[r]^{s^-}(0)}\frac{1}{D[g]^{s^+}\circ r(0)}.$$

Now, when a_1/a_0 varies in the set of all inputs of Ω with a given height p, we obtain

$$\sum_{\substack{(a_0,a_1)\in\Omega\\ P(a_0,a_1)=p}} \frac{a_i^\gamma}{a_0^{\alpha\gamma+s}} = \mathbf{H}_{s^-}^{p-i}\circ \mathbf{H}_{s^+}^{i}[f](0).$$

Finally, a summation over p gives the result. Moreover, the relation $\widetilde{F}_M(s) = \zeta(s^+)F_M(s)$ proves that it is sufficient to work with inputs of Ω. ∎

4.2. **Dominant singularity of $F_M(s)$.** With the notations of Figure 7, the expression of the previous Lemma takes the following form,

$$F_M(s) = \sum_{j=0}^{c+d-1} \mathbf{H}_{s^-}^{j-j'} \circ \left(\sum_{k\geq 0} \mathbf{H}_{s^-}^{dk}\circ \mathbf{H}_{s^+}^{ck}\right)\circ \mathbf{H}_{s^+}^{j'}[f](0), \tag{4.2}$$

and the possible singularities of the series $F_M(s)$ become apparent.

Lemma 4. *Let s be a real $s > 1$, and let $R(s)$ be the spectral radius of the operator $\mathbf{H}_s$. Denote by $\phi(s)$ the function*

$$\phi(s) := R^d(s^-)\ R^c(s^+). \tag{4.3}$$

The equation $\phi(s) = 1$ has a unique real solution ρ. For $0 < \gamma < 1$, this solution belongs to the interval $[2-\alpha\gamma, 2+\beta\gamma]$.
The function $F_M(s)$ is analytic on the half-plane $\Re(s) > \rho$.

Proof. The function ϕ is defined for $s > 1+\beta\gamma$. Notice that if the solution ρ exists, one has $\rho^- < 2 < \rho^+$. Moreover, for $0<\gamma<1$, one has $2-\alpha\gamma > 1+\beta\gamma$ so that it is sufficient to study $\phi(s)$ on the interval

$$[2-\alpha\gamma, 2+\beta\gamma]. \tag{4.4}$$

Since $s \to R(s)$ is strictly decreasing on the real axis, the same is true for $s \to \phi(s)$ (on the real axis), and the sequence of inequalities

$$\phi(2-\alpha\gamma) = R^d(2-\gamma) > \lambda(2) = 1 > R^c(2+\gamma) = \phi(2+\beta\gamma).$$

proves that the equation $\phi(s) = 1$ has a unique solution ρ.

From (4.2), one obtains, when taking norms on space $BV(I)$,

$$|F_M(s)| \leq ||f|| \left(\sum_{j=0}^{c+d-1} ||\mathbf{H}_{s^-}^{j-j'}|| \, ||\mathbf{H}_{s^+}^{j'}|| \right) \left(\sum_{k\geq 0} ||\mathbf{H}_{s^-}^{dk}|| \, ||\mathbf{H}_{s^+}^{ck}|| \right).$$

The right part defines a series whose general term is equivalent to $R(s^-)^{dk} R(s^+)^{ck}$. This series is convergent when $\phi(s) = R(s^-)^d R(s^+)^c$ is less than one. ∎

Lemma 5. *Suppose that γ is positive and sufficiently small. Then, the series $F_M(s)$ has a pole of order 1 at $s = \rho$ where ρ is defined as the unique solution of the equation $\phi(s) = 1$. Moreover, $F_M(s)$ is analytic on the half–plane $\{\Re(s) \geq \rho, s \neq \rho\}$.*

Proof. If γ is sufficiently small, the interval $]2-\gamma, 2+\gamma[$ is contained in the neighborhood $\mathcal{V}$ of Section 3.6. Then, when s belongs to the interval defined by (4.4), the two values s^+ and s^- belong to $\mathcal{V}$, and the two operators $\mathbf{H}_{s^+}, \mathbf{H}_{s^-}$ are quasi–compact. Then, the dominant eigenvalue $\lambda(s^+), \lambda(s^-)$ of the operators $\mathbf{H}_{s^+}, \mathbf{H}_{s^-}$ are well–defined. The spectral decomposition (3.13) of $\mathbf{H}_t$ described in Section 3 applies to $t = s^+, t = s^-$ and extends to the series $F_M(s)$ via the equality (4.2) so that, on the interval (4.4), the series $F_M(s)$ decomposes into a sum of two terms, a "dominant" term and a "remainder" term. The dominant term is obtained when replacing each occurrence of $\mathbf{H}_t$ by the term $\lambda(t)\mathbf{P}_t$, and is of the form $F_M^+(s)\,\mathbf{P}_{s^-} \circ \mathbf{P}_{s^+}[f](0)$, with

$$\begin{aligned} F_M^+(s) &= \left(\sum_{j=0}^{c+d-1} \lambda^{j-j'}(s^-)\lambda^{j'}(s^+) \right) \left(\sum_{k\geq 0} (\lambda^d(s^-)\lambda^c(s^+))^k \right), \\ &= \frac{1}{1-\phi(s)} \left(\sum_{j=0}^{c+d-1} \lambda^{j-j'}(s^-)\lambda^{j'}(s^+) \right). \end{aligned} \tag{4.5}$$

Spectral properties of the operator $\mathbf{H}_t$, for $t = s^+, t = s^-$, prove that the dominant poles of the series $F_M(s)$ are only brought by the dominant part $F_M^+(s)$. Thus, the dominant pole of $F_M(s)$ arises at $s = \rho$ if ρ is solution of the equation $\phi(s) = 1$. Near $s = \rho$, one has

$$F_M(s) \sim \frac{1}{s-\rho} \, \frac{-1}{\phi'(\rho)} \sum_{j=0}^{c+d-1} \lambda^{j-j'}(\rho^-)\lambda^{j'}(\rho^+) \mathbf{P}_{\rho^-} \circ \mathbf{P}_{\rho^+}[f](0). \tag{4.6}$$

The maximum properties of the function $s \to \lambda(s)$ along vertical and horizontal lines are inherited by $s \to \phi(s)$, so that $\phi(s) < 1$ for $\Re(s) = \rho, s \neq \rho$. ∎

Then, the series $F_M(s)$ fullfills all the conditions of Tauberian theorem with $\sigma = \rho$ and $\theta = 0$ when γ is positive. Remark that the function ϕ is the same for the two pairs (α, γ) and $(1-\alpha, -\gamma)$. So, all the previous results are also true for γ negative, sufficiently close to 0. Since the Dirichlet series $F_1(s)$ fullfills all the conditions of Tauberian theorem with $\sigma = 2$ and $\theta = 0$ (as proven in Section 3.7), this leads to the main result of this Section.

Proposition 2. *For all γ sufficiently close to 0, for all $\alpha, \delta \in [0,1]$, there exists ρ (that depends on α, γ and δ) such that the expectation of the cost M satisfies*

$$\mathrm{E}_n\left[\left(\frac{a_{\lfloor \delta p \rfloor}}{a_0^{\alpha}}\right)^{\gamma}\right] = O(2^{n(\rho-2)}).$$

4.3. **Particular case when δ is near $1-\alpha$.** We now prove that ρ is less than 2 when δ equals $(1-\alpha)+\varepsilon$ with $\varepsilon > 0$.

Lemma 6. *When $\delta = (1-\alpha)+\varepsilon$, with $\varepsilon > 0$ (resp. $\varepsilon < 0$), there exists $\gamma > 0$ (resp. $\gamma < 0$) such that the unique solution ρ of Equation $\phi(s) = 1$ is strictly less than 2. Moreover, one can choose $2-\rho = \Omega(\varepsilon^2)$.*

Proof. Suppose that $\delta = \frac{c}{c+d}$ is of the form $\delta = (1-\alpha)+\varepsilon$. One has

$$\frac{c}{d} = \frac{1-\alpha+\varepsilon}{\alpha-\varepsilon} = \frac{\beta+\varepsilon}{\alpha-\varepsilon}.$$

Now, equation $\phi(s) = 1$ can be written with the fonction $\Lambda(s) := \log \lambda(s)$ as

$$\Phi(s) := \frac{|\Lambda(s^-)|}{|\Lambda(s^+)|} = \frac{c}{d} = \frac{\beta+\varepsilon}{\alpha-\varepsilon}.$$

On a neighborhood of $s = 2$, the left term defines a strictly decreasing function Φ of s. It is thus sufficient to show that there exists some $\gamma > 0$ (that will depend on ε) for which

$$\Phi(2) < \Phi(\rho) = \frac{\beta+\varepsilon}{\alpha-\varepsilon}.$$

The function $s \to \Lambda(s)$ satisfies the following

$$\Lambda(2) = 0, \qquad \Lambda'(2) < 0, \qquad \Lambda''(2) > 0,$$

so that, for a sufficiently small γ, one has

$$\Phi(2) = \frac{|\Lambda(2-\beta\gamma)|}{|\Lambda(2+\alpha\gamma)|} < \frac{\beta}{\alpha}\,\frac{1+\beta\varepsilon_1}{1-\alpha\varepsilon_1} < \frac{\beta+\varepsilon_1}{\alpha-\varepsilon_1}, \qquad \text{with} \qquad \varepsilon_1 := \frac{3\gamma}{4}\,\frac{\Lambda''(2)}{|\Lambda'(2)|}.$$

One can choose

$$\gamma = \varepsilon\frac{|\Lambda'(2)|}{\Lambda''(2)} \qquad \text{so that} \qquad \varepsilon_1 = \frac{3\varepsilon}{4} < \varepsilon \qquad \text{and } \Phi(2) < \Phi(\rho).$$

In this case, one has $\rho < 2$, and this proves the first part of the Lemma. We now wish to evaluate $2-\rho$ as a function of ε. First

$$2-\rho \sim \frac{|\Phi(2)-\Phi(\rho)|}{\Phi'(2)}.$$

Then,

$$|\Phi(2)-\Phi(\rho)| \geq \frac{\beta+\varepsilon}{\alpha-\varepsilon} - \frac{\beta+\varepsilon_1}{\alpha-\varepsilon_1} \geq \frac{\varepsilon-\varepsilon_1}{\alpha^2} = \frac{\varepsilon}{4\alpha^2}.$$

On the otherside, when taking logarithmic derivatives, and using the fact that $|(\Lambda'/\Lambda)(2+x)| \sim (1/x)$ (near $x=2$), one has

$$\frac{|\Phi'(2)|}{\Phi(2)} = \left|\frac{\Lambda'}{\Lambda}(2-\beta\gamma)\right| + \left|\frac{\Lambda'}{\Lambda}(2+\alpha\gamma)\right| \qquad \text{so that} \qquad |\Phi'(2)| \sim \frac{1}{\alpha^2\gamma}$$

and finally

$$2-\rho \geq \frac{\varepsilon\gamma}{4} = \frac{\varepsilon^2}{4}\frac{|\Lambda'(2)|}{\Lambda''(2)}. \qquad \blacksquare$$

This is the end of the proof of Theorem 1. Part (i). We prove now the second part of Theorem 1.

4.4. **Proof of Theorem 1. Part** (ii). Denote by Q_α the random variable $Q_\alpha := |P_\alpha - (1-\alpha)P|$, and consider, for some $\varepsilon > 0$, the exceptional event $A(\varepsilon) := [Q_\alpha \geq \varepsilon P]$. The worst case of the Euclidean algorithm entails that, on Ω_n, one has always $P = O(n)$. Furthermore, the relation

$$\mathrm{E}_n[Q_\alpha] \leq Kn\ \left(\Pr_n[A(\varepsilon)] + \varepsilon\right),$$

together with Part (i) of Theorem 1 proves that

$$\mathrm{E}_n[P_\alpha] \sim (1-\alpha)\mathrm{E}_n[P],$$

and this ends the proof of Theorem 1.

5. Analysis of the Interrupted Algorithm. Bit complexity.

In this section, we prove Theorem 2. We are interested in studying the average bit-complexity of the Interrupted Euclidean algorithm $\mathcal{E}_\alpha$ which terminates as soon as the current integer a_i is less than a_0^α. As claimed in Section 3.2, we replace this algorithm by a "regularized" algorithm $\overline{\mathcal{E}}_\alpha$ which always terminates at a fraction of the depth. Thanks to Theorem 1, the two algorithms are quite close, and their bit-complexities will be asymptotically the same.

5.1. **Expression of Dirichlet series with transfer operators.** As in the previous section, the first step relates the Dirichlet series of costs $\overline{B}_\alpha, \overline{C}_\alpha$,

$$(5.1) \quad \overline{B}_\alpha(a_0,a_1) = \sum_{i=1}^{\lfloor(1-\alpha)p\rfloor} \ell(a_i)\cdot b(q_i) \qquad \text{with} \qquad b(q) := D\ell(q) + 2M$$

$$(5.2) \quad \overline{C}_\alpha(a_0,a_1) = \sum_{i=1}^{\lfloor(1-\alpha)p\rfloor} \ell(v_i)\cdot c(q_i) \qquad \text{with} \qquad c(q) := (\ell(q)+2)M$$

to some transfer operators. These costs involve different parameters, namely the length $\ell(q)$ of the quotients q relative to the LFT $h : h(x) = 1/(q+x)$, the length $\ell(a_i)$ of the remainders, and the lengths of the Bezout terms u_i, v_i. As in (3.9,3.10), we consider an input (a_0,a_1) of the algorithm such that $a_1/a_0 = h(0) = h_1 \circ h_2 \circ \cdots \circ h_p(0)$ and we split the LFT h in three parts :

(a) the beginning part $b_i(h) = h_1 \circ \cdots \circ h_{i-1}$,
(b) the ending part $e_i(h) = h_{i+1} \circ \cdots \circ h_p$
(c) and the i-th component h_i.

Then, the operator relative to the LFT h decomposes as

$$\mathbf{R}_{s,h} = \mathbf{R}_{s,e_i(h)} \circ \mathbf{R}_{s,h_i} \circ \mathbf{R}_{s,b_i(h)}.$$

First, we weight the operator $\mathbf{R}_{s,h_i}$ with some cost $d(q_i)$ relative to the quotient q_i of the LFT h_i and introduce, for any LFT g of depth 1, the operator

$$\mathbf{R}^{[d]}_{s,g}[f](x) := \frac{d(q)}{D[g](x)^s}\, f \circ g(x).$$

Second, we take derivatives with respect to s, and use the derivative functional Δ that is defined for an operator $\mathbf{L}_s$ that depends on parameter s by

$$\Delta \mathbf{L}_s := -\frac{1}{\log 2}\frac{d}{ds}\mathbf{L}_s.$$

When applied to $\mathbf{R}_{s,g}$, it produces at the numerator the logarithm $\log_2 D[g]$. Since $D[e_i(h)](0) = a_i$, $D[b_{i-1}(h)](x) = v_i + v_{i-1}x$, we obtain, with costs b, c defined in (5.1, 5.2),

$$\Delta \mathbf{R}_{s,e_i(h)} \circ \mathbf{R}^{[b]}_{s,h_i} \circ \mathbf{R}_{s,b_i(h)}[f](0) = \frac{1}{a_0^s}\log_2 a_i\, b(q_i) f(\frac{a_1}{a_0}),$$

$$\mathbf{R}_{s,e_i(h)} \circ \mathbf{R}^{[c]}_{s,h_i} \circ \Delta\mathbf{R}_{s,b_{i-1}(h)}[f](0) = \frac{1}{a_0^s}\log_2\left(v_i + v_{i-1}\frac{a_i}{a_{i-1}}\right) c(q_i)\, f(\frac{a_1}{a_0}).$$

The inequalities

$$|\log_2 a_i - \ell(a_i)| \le 1, \qquad |\log_2\left(v_i + v_{i-1}\frac{a_i}{a_{i-1}}\right) - \ell(v_i)| \le 1 \tag{5.3}$$

prove that the costs $\widehat{B}_\alpha, \widehat{C}_\alpha$ defined as

$$\widehat{B}_\alpha(a_0, a_1) = \sum_{i=1}^{\lfloor(1-\alpha)p\rfloor} \log_2(a_i)\cdot b(q_i), \tag{5.4}$$

$$\widehat{C}_\alpha(a_0, a_1) = \sum_{i=1}^{\lfloor(1-\alpha)p\rfloor} \log_2\left(v_i + v_{i-1}\frac{a_i}{a_{i-1}}\right)\cdot c(q_i), \tag{5.5}$$

can be viewed as "approximations" for costs $\overline{B}_\alpha, \overline{C}_\alpha$ (We shall prove this statement in a precise way in Section 5.3). Now, the operators

$$\mathbf{B}^{(\alpha)}_{s,h} = \sum_{i=0}^{\lfloor(1-\alpha)p\rfloor} \Delta\mathbf{R}_{s,e_i(h)} \circ \mathbf{R}^{[b]}_{s,h_i} \circ \mathbf{R}_{s,b_i(h)},$$

$$\mathbf{C}^{(\alpha)}_{s,h} = \sum_{i=0}^{\lfloor(1-\alpha)p\rfloor} \mathbf{R}_{s,e_i(h)} \circ \mathbf{R}^{[c]}_{s,h_i} \circ \Delta\mathbf{R}_{s,b_i(h)},$$

are generating operators for the costs $\widehat{B_\alpha}$ and $\widehat{C_\alpha}$ on the input (a_0, a_1), since

$$\mathbf{B}^{(\alpha)}_{s,h}[f](0) = \frac{1}{a_0^s}\widehat{B}_\alpha(a_0, a_1) f(\frac{a_1}{a_0}), \qquad \mathbf{C}^{(\alpha)}_{s,h}[f](0) = \frac{1}{a_0^s}\widehat{C}_\alpha(a_0, a_1) f(\frac{a_1}{a_0}).$$

When (a_0, a_1) is a generic element of Ω with height p, the LFT h is a generic element of $\mathcal{H}^p$. When summing over p, we obtain:

Lemma 7. *The Dirichlet series $F_X(s)$ relative to costs $X = \widehat{B_\alpha}$ or $X = \widehat{C_\alpha}$ defined in (5.4, 5.5) are expressed in terms of the transfer operator $\mathbf{H}_s$, the* *weighted operator* $\mathbf{H}_s^{[d]}$ *relative to cost* $d = b, c$ *defined in (5.1, 5.2), and the functional derivative* Δ,

$$F_{\widehat{B}_\alpha}(s) = \sum_{p\geq 0} \sum_{i=1}^{\lfloor(1-\alpha)p\rfloor} \Delta\mathbf{H}_s^{p-i} \circ \mathbf{H}_s^{[b]} \circ \mathbf{H}_s^{i-1}[f](0),$$

$$F_{\widehat{C}_\alpha}(s) = \sum_{p\geq 0} \sum_{i=1}^{\lfloor(1-\alpha)p\rfloor} \mathbf{H}_s^{p-i} \circ \mathbf{H}_s^{[c]} \circ \Delta\mathbf{H}_s^{i-1}[f](0).$$

5.2. **Spectral decomposition and Tauberian Theorem.** We now work with s near to 2 and use the spectral decomposition (3.13) which splits the operator $\mathbf{H}_s^n$ in two parts

$$\mathbf{H}_s^n = \lambda^n(s)\mathbf{P}_s + \mathbf{N}_s^n$$

where $\lambda(s)$ is the dominant eigenvalue of the operator. This decomposition extends to the series. There appears a "dominant" term that is obtained when replacing all the powers $\mathbf{H}_s^n$ by $\lambda(s)^n\mathbf{P}_s$. Remark also that

$$\Delta\mathbf{H}_s^n = \sum_{j=1}^{n} \mathbf{H}_s^{j-1} \circ \Delta\mathbf{H}_s \circ \mathbf{H}_s^{n-j}$$

brings a dominant term equal to $n\lambda^{n-1}(s)\mathbf{P}_s \circ \Delta\mathbf{H}_s \circ \mathbf{P}_s$. Finally, the dominant terms $F^+_{\widehat{B}_\alpha}(s), F^+_{\widehat{C}_\alpha}(s)$ of $F_{\widehat{B}_\alpha}(s), F_{\widehat{C}_\alpha}(s)$ involve the operators

$$\mathbf{Q}_s = \mathbf{P}_s \circ \Delta\mathbf{H}_s \circ \mathbf{P}_s \circ \mathbf{H}_s^{[b]} \circ \mathbf{P}_s, \qquad \mathbf{T}_s = \mathbf{P}_s \circ \mathbf{H}_s^{[c]} \circ \mathbf{P}_s \circ \Delta\mathbf{H}_s \circ \mathbf{P}_s$$

under the form

$$F^+_{\widehat{B}_\alpha}(s) = \widehat{B}^{(\alpha)}(s)\ \mathbf{Q}_s[f](0), \qquad F^+_{\widehat{C}_\alpha}(s) = \widehat{C}^{(\alpha)}(s)\ \mathbf{T}_s[f](0)$$

with (near $s = 2$)

$$\widehat{B}^{(\alpha)}(s) = \sum_{p\geq 1} \left(\sum_{i=0}^{\lfloor(1-\alpha)p\rfloor} (p-i) \right) \lambda^{p-1}(s) \sim (1-\alpha^2) \left(\frac{1}{1-\lambda(s)}\right)^3,$$

$$\widehat{C}^{(\alpha)}(s) = \sum_{p\geq 1} \left(\sum_{i=0}^{\lfloor(1-\alpha)p\rfloor} i \right) \lambda^{p-1}(s) \sim (1-\alpha)^2 \left(\frac{1}{1-\lambda(s)}\right)^3.$$

Near $s = 2$, the relations

$$\frac{\widehat{B}^{(\alpha)}(s)}{\widehat{B}^{(0)}(s)} \sim (1-\alpha^2), \qquad \frac{\widehat{C}^{(\alpha)}(s)}{\widehat{C}^{(0)}(s)} \sim (1-\alpha)^2$$

prove an analog of Theorem 2 when the bar is replaced by a hat, namely,

$$\mathrm{E}_n[\widehat{B}_\alpha] \sim (1-\alpha^2)\, \mathrm{E}_n[\widehat{B}_0], \qquad \mathrm{E}_n[\widehat{C}_\alpha] \sim (1-\alpha)^2\, \mathrm{E}_n[\widehat{C}_0] \tag{5.6}$$

for $n \to \infty$.

Furthermore, the asymptotic behaviour of the expectations $\mathrm{E}_n[\widehat{B}_0], \mathrm{E}_n[\widehat{C}_0]$ can be obtained when applying Tauberian Theorem as in [1, 27]: Near $s = 2$, the two quantities $\mathbf{Q}_s[f](0), \mathbf{T}_s[f](0)$ define analytic functions of s whose values at $s = 2$ involve two integrals

$$\int_0^1 \mathbf{H}^{[d]}[\psi](t)dt, \qquad \int_0^1 \Delta\mathbf{H}[\psi](t)dt \tag{5.7}$$

which deal with the stationary density ψ, and the possible costs $d = b, c$. As in [1, 27], one has (we recall that ℓ is the binary length)

$$\int_0^1 \mathbf{H}^{[\ell]}[\psi](t)dt = \log_2 \prod_{k=0}^{\infty}(1+\frac{1}{2^k}), \qquad \int_0^1 \Delta\mathbf{H}[\psi](t)dt = -\lambda'(2)$$

so that

$$\mathrm{E}_n[\widehat{B}_0] \sim (L_1 M + L_2 D)\, n^2, \qquad \mathrm{E}_n[\widehat{C}_0] \sim (L_1 + L_2) M\, n^2$$

$$\text{with} \qquad L_1 = \frac{12\log^2 2}{\pi^2}, \qquad L_2 = \frac{6\log 2}{\pi^2}\log\prod_{k=0}^{\infty}(1+\frac{1}{2^k}).$$

We now come back to the original costs, first with a bar, second without a bar.

5.3. **Relation between costs with a hat and costs with a bar.** The Dirichlet series of the cost $Y(a_0, a_1) := \sum_{i=1}^{p} d(q_i)$ corresponding to $d = b, c$ is exactly

$$(I - \mathbf{H}_s)^{-1} \circ \mathbf{H}_s^{[d]} \circ (I - \mathbf{H}_s)^{-1}[f](0).$$

Since this series has a pole of order 2 at $s = 2$, the expectation $\mathrm{E}_n[Y]$ is $O(n)$. Finally, the inequalities (5.3) entail that

$$\mathrm{E}_n[\widehat{B}_\alpha] - E_n[\overline{B}_\alpha] = O(n), \qquad \mathrm{E}_n[\widehat{C}_\alpha] - E_n[\overline{C}_\alpha] = O(n).$$

5.4. **Relation between the costs with a bar and without a bar.** In the same vein as in Section 4.4, denote by Q_α the random variable $Q_\alpha := |P_\alpha - (1-\alpha)P|$ and by R_α one of the two random variables $R_\alpha := |B_\alpha - \overline{B_\alpha}|$ or $R_\alpha := |C_\alpha - \overline{C_\alpha}|$. The worst case of the Euclidean algorithm entails that, on Ω_n, one has always $R_\alpha = O(n^2)$. Furthermore, when dealing with the exceptional event $A(\varepsilon) := [Q_\alpha \geq \varepsilon P]$, one obtains the relation

$$\mathrm{E}_n[R_\alpha] \leq K'\, n^2\, \mathrm{Pr}_n\,[A(\varepsilon)] + n\,\mathrm{E}_n\,[X_\alpha] \qquad \text{with} \qquad X_\alpha := \sum_{i=\lfloor(1-\alpha-\varepsilon)P\rfloor}^{\lfloor(1-\alpha+\varepsilon)P\rfloor} d(q_i), \tag{5.8}$$

where d is one of the two costs b or c. Theorem 1 proves that the first term is $o(n^2)$. The Dirichlet series relative to cost X_α,

$$\sum_{p\geq 0}\sum_{i=\lfloor(1-\alpha-\varepsilon)p\rfloor}^{\lfloor(1-\alpha+\varepsilon)p\rfloor} \mathbf{H}_s^{p-i} \circ \mathbf{H}_s^{[d]} \circ \mathbf{H}_s^{i-1}[f](0),$$

has a dominant term equal to

$$\varepsilon\left(\sum_p p\lambda(s)^{p-1}\right)\mathbf{P}_s \circ \mathbf{H}_s^{[d]} \circ \mathbf{P}_s[f](0) = \varepsilon\left(\frac{1}{1-\lambda(s)}\right)^2 \mathbf{P}_s \circ \mathbf{H}_s^{[d]} \circ \mathbf{P}_s[f](0),$$

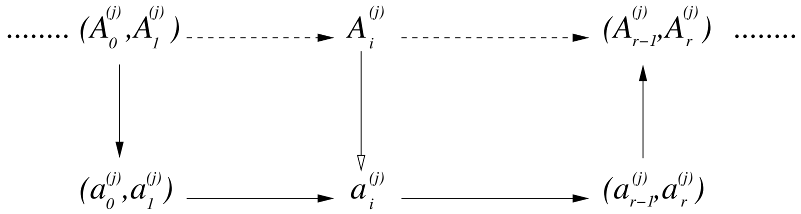

FIGURE 8. The j–th phase of the Lehmer-Euclid algorithm.

so that the second term of (5.8) is of the form $o(n^2)$. Finally, for both costs R_α, one has $\mathrm{E}_n[R_\alpha] = o(n^2)$, and

$$\mathrm{E}_n[B_\alpha] \sim \mathrm{E}_n[\overline{B_\alpha}], \qquad \mathrm{E}_n[C_\alpha] \sim \mathrm{E}_n[\overline{C_\alpha}].$$

Finally, Theorem 2 is obtained with (5.6), Sections 5.3 and 5.4.

6. Analysis of the Lehmer-Euclid Algorithm.

We now prove Theorem 3. There are two mains ideas, summarized by the next two lemmas. The first lemma compares, during each phase, the quantities that would appear if one used the usual Euclidean Algorithm (the top horizontal line of Figure 8) with the quantities that actually appear in the bottom horizontal line of Figure 8.

Since these quantities are closely related , we can "simulate" the bit-complexity of the Lehmer–Euclid algorithm only on the plain Euclid algorithm itself. The second lemma shows that the algorithm is almost surely "regular" in the sense that the duration of each phase is (almost surely) equal to a fraction of the depth. Then, as in the previous section, we first study, in the third lemma, a regularized version of the Lehmer-Euclid Algorithm $\mathcal{LE}_\mu$, that we denote by $\overline{\mathcal{LE}}_\mu$, where the duration of each phase is exactly equal to a fraction of the depth, i.e., $\lfloor (\mu/2)P \rfloor$. This Section ends when comparing the bit–complexity of the two algorithms, the $\mathcal{LE}_\mu$ algorithm and the $\overline{\mathcal{LE}}_\mu$ algorithm.

In the top horizontal line, the j-th phase begins with the pair $(A_0^{(j)}, A_1^{(j)})$ while the bottom horizontal line starts with

$$(a_0^{(j)}, a_1^{(j)}) := T_m(A_0^{(j)}, A_1^{(j)}).$$

At the i-th step of the j-th phase, the Lehmer-Euclid algorithm deals with the pair $(a_{i-1}^{(j)}, a_i^{(j)})$ while the Euclid algorithm (if it was performed) would deal with the pair $(A_{i-1}^{(j)}, A_i^{(j)})$. Except at the first step of the j–th phase, it is not true that the small pair $(a_{i-1}^{(j)}, a_i^{(j)})$ is the truncation of the large pair $(A_{i-1}^{(j)}, A_i^{(j)})$: generally speaking, the two pairs $(a_{i-1}^{(j)}, a_i^{(j)})$ and $T_m(A_{i-1}^{(j)}, A_i^{(j)})$ are not equal. However, the next lemma shows that these pairs have almost the same length.

Lemma 8. *Denote by $(a_{i-1}^{(j)}, a_i^{(j)})$ the pair that is used by the Lehmer-Euclid Algorithm at the i-th step of the j-th phase, and by $(A_{i-1}^{(j)}, A_i^{(j)})$ the pair that* *would be used by the Euclid Algorithm at the corresponding step. Then, one has*

$$|\ell(a_i^{(j)}) - \ell(A_i^{(j)}) + \ell(A_0^{(j)}) - m| \leq 2.$$

Proof. We fix a phase, and we omit the index j. We use the notations of Section 2.1. Since the two matrices $\mathcal{M}_i$ (the "large" one and the "small" one) are equal, we have

$$(6.1) \quad \begin{pmatrix} A_{i+1} \\ A_i \end{pmatrix} = \mathcal{M}_i^{-1} \begin{pmatrix} A_1 \\ A_0 \end{pmatrix} \qquad \text{and} \qquad \begin{pmatrix} a_{i+1} \\ a_i \end{pmatrix} = \mathcal{M}_i^{-1} \begin{pmatrix} a_1 \\ a_0 \end{pmatrix}.$$

On the otherside, since the pair (a_0, a_1) is the m–truncation of pair (A_0, A_1),

$$\begin{pmatrix} A_1 \\ A_0 \end{pmatrix} = 2^{\ell(A_0)-m} \left[\begin{pmatrix} a_1 \\ a_0 \end{pmatrix} + \begin{pmatrix} \delta_1 \\ \delta_0 \end{pmatrix} \right],$$

where δ_0, δ_1 satisfy $0 \leq \delta_0, \delta_1 < 1$. Then,

$$\begin{pmatrix} A_{i+1} \\ A_i \end{pmatrix} = 2^{\ell(A_0)-m} \left[\begin{pmatrix} a_{i+1} \\ a_i \end{pmatrix} + \mathcal{M}_i^{-1} \begin{pmatrix} \delta_1 \\ \delta_0 \end{pmatrix} \right].$$

Moreover, during each step, the relation $a_0 = |v_{i-1}|\, a_i + |v_i|\, a_{i-1}$ together with the inequalities $a_i, a_{i-1} > \sqrt{a_0}$ prove that the absolute values of all the coefficients of matrix $\mathcal{M}_{i-1}, \mathcal{M}_{i-1}^{-1}$ are less than $(1/2)\sqrt{a_0} \leq (1/2) a_i$. Finally, the relation

$$|A_i - 2^{\ell(A_0)-m} a_i| \leq 2^{\ell(A_0)-m}\, \frac{a_i}{2}$$

proves the Lemma. ■

The bit–cost of the j–th phase decomposes into three bit–costs. The first one (type 1) is the bit-cost of an Interrupted Euclidean Algorithm, the second one (type 2) is the extra cost due to the computation of the two cosequences, and the third one (type 3) is due to the four multiplications of Stage 3. The second and the third costs (type 2 and 3) involve quantities that could have been computed during the Euclidean Algorithm of the top horizontal line, so that the Dirichlet series of these two costs admit expressions that involve the transfer operators relative to the Euclidean Dynamical system, with a possible intervention of the cost $d = c$ or $d = b$, and of the functional Δ. The first cost (type 1) involves the quantities previously denoted by $\ell(a_i^{(j)})$ that are not computed by the "top" Euclidean algorithm. However, Lemma 8 proves that these quantities can be approximated by other quantities $\ell(A_0^{(j)}), \ell(A_i^{(j)})$ that are computed by the "top" Euclidean algorithm.

We denote by J the number of phases, by $p(j)$ the beginning index of the j–th phase, by $v[r,t]$ the coefficient v that is computed between the two indices $i = r$ and $i = t$. Here are the different costs involved during the algorithm:

Array 1. *Costs.*

Cost of type 1: $\sum_{j=1}^{J} \sum_{i=p(j)}^{p(j+1)-1} [\ell(A_i) - \ell(A_{p(j)}) + m] \cdot b(q_i)$

Cost of type 2: $2 \sum_{j=1}^{J} \sum_{i=p(j)}^{p(j+1)-1} \ell(v[p(j), i]) \cdot c(q_i)$

Cost of type 3: $4 \sum_{j=1}^{J} \ell(A_{p(j)}) \cdot \ell(v[p(j), p(j+1)])$

In fact, the cost of type 1 decomposes into three different costs. The first one gives rise to the bit-complexity of the plain Euclidean Algorithm during the phase and the third one was analyzed in Section 5.3. It remains the second part, that we call cost of type 1':

$$\text{Cost of type 1': } \sum_{j=1}^{J} \ell(A_{p(j)}) \cdot \sum_{i=p(j)}^{p(j+1)-1} b(q_i)$$

All the previous expressions heavily depend on the sequence of indices $p(j)$ that denote the beginning of the j-th phase. The next Lemma proves that the length of each phase is almost surely the expected one, namely equal to $\bar{p} := \lfloor (\mu/2)p \rfloor$, so that the sequence $p(j)$ is not too far from an arithmetic progression. This Lemma is an (easy) extension of Theorem 1.

Lemma 9. *Denote by $\delta(j)$ the number of iterations that are performed during the j–th phase. For any ε, there exists $K < 1$ for which, when $n \to \infty$,*

$$\Pr_n \left[\left| \frac{\delta(j)}{P} - \frac{\mu}{2} \right| > \varepsilon \right] = O(K^n).$$

Proof. If a phase begins at the r–th iteration of the Euclidean algorithm, it ends at the $(r+t)$–th iteration, as soon as the (small) sequence a_i satisfies some condition. However, the previous lemma proves that there is a precise relation between the length of small a_i and the length of large A_i. In fact, the phase ends as soon as

$$A_{r+t} \sim \frac{A_r}{A_0^{\mu/2}}.$$

As in the proof of Theorem 1, we use Markov's inequality and we are led to study the Dirichlet series relative to the cost

$$N(A_0, A_1) := \left(\frac{A_{r+t}}{A_r A_0^{-\mu/2}} \right)^{\gamma}$$

for some $\gamma > 0$. We use the same notations as in Figure 7 with $\alpha = 1 - (\mu/2)$ and $\beta = \mu/2$. This Dirichlet series admits an alternative expression that involves the transfer operator $\mathbf{H}_s$, as

$$\mathbf{H}_{s^-}^{p-r-t} \circ \mathbf{H}_{s^+}^{t} \circ \mathbf{H}_{s^-}^{r}.$$

It is important to remark that the dominant part of the function does not depend on the index r that denotes the moment when the phase begins. If we let $p := (c+d)k, t := ck$, the function ϕ that is relative to the dominant singularity is exactly the same as in Lemma , i.e., $\phi(s) := \lambda^d(s^-)\lambda^c(s^+)$. The end of the proof is exactly the same as in Section 4. ∎

Since the length of each phase is almost surely equal to $\bar{p} := \lfloor (\mu/2)p \rfloor$, we now consider the regularized version of the Lehmer-Euclid Algorithm, where the length of each phase equals exactly $\bar{p} := \lfloor (\mu/2)p \rfloor$. We denote this algorithm by $\overline{\mathcal{LE}}_\mu$. We shall prove two facts. First, the expectation of the bit–cost of the $\overline{\mathcal{LE}}_\mu$ algorithm on $\Omega_n, \widetilde{\Omega}_n$ is given by the expression of Theorem 3. Second, the difference between the two average bit–costs, the average bit–cost of the $\overline{\mathcal{LE}}_\mu$ algorithm on $\Omega_n, \widetilde{\Omega}_n$ and the average bit–cost of the $\mathcal{LE}_\mu$

algorithm on $\Omega_n, \widetilde{\Omega}_n$ is $o(n^2)$. We first consider the regularized version of the Algorithm.

Lemma 10. *The expectation of the bit-cost during the j-th phase of the $\overline{\mathcal{LE}}_\mu$ algorithm on $\Omega_n, \widetilde{\Omega}_n$ is asymptotically equal to*

$$\frac{3}{4}(L_1 M + L_2 D)\mu^2 n^2 + \frac{1}{2}(L_1 + L_2) M \mu^2 n^2 + 4\frac{\mu}{2}(1 - (j-1)\frac{\mu}{2}) M n^2.$$

Proof. We first obtain an expression for the costs of the $\overline{\mathcal{LE}}_\mu$ algorithm during the j-th phase when the depth of the algorithm equals p.

Array 2. *Costs of the Regularized Algorithm during the j-th phase.*

Cost of type 1': $\sum_{i=j\bar{p}}^{(j+1)\bar{p}-1} \ell(A_{j\bar{p}}) \cdot b(q_i)$

Cost of type 2: $2\sum_{i=j\bar{p}}^{(j+1)\bar{p}-1} \ell(v[j\bar{p}, i]) \cdot c(q_i)$

Cost of type 3: $4\ell(A_{j\bar{p}}) \cdot \ell(v[j\bar{p}, (j+1)\bar{p}])$

We easily deduce an expression for the Dirichlet series of "regularized" costs when the depth equals p. In fact, we work with some approximate costs that are the analogs of the costs with a hat of the previous Section.

Array 3. *Dirichlet series for costs of the regularized algorithm during the j-th phase.*

Cost of type 1': $\sum_{i=1}^{\bar{p}} \Delta \left[\mathbf{H}_s^{p-j\bar{p}} \circ \mathbf{H}_s^{\bar{p}-i} \circ \mathbf{H}_s^{[b]} \circ \mathbf{H}_s^{i-1}\right] \circ \mathbf{H}_s^{(j-1)\bar{p}}[f](0)$

Cost of type 2: $\sum_{i=1}^{\bar{p}} \mathbf{H}_s^{p-j\bar{p}} \circ \mathbf{H}_s^{\bar{p}-i} \circ \mathbf{H}_s^{[c]} \circ \Delta \left[\mathbf{H}_s^{i-1}\right] \circ \mathbf{H}_s^{(j-1)\bar{p}}[f](0)$

Cost of type 3: $4\Delta \left[\mathbf{H}_s^{p-j\bar{p}} \circ \Delta\left(\mathbf{H}_s^{\bar{p}}\right)\right] \circ \mathbf{H}_s^{(j-1)\bar{p}}[f](0)$

When summing over all possible values of depth p, the dominant terms of the Dirichlet series of Array 3 all involve a product of two factors. The first factor is the same for all the three series and is equal to

$$\sum_{p \geq 0} p^2 \lambda(s)^p \sim 2\left(\frac{1}{1-\lambda(s)}\right)^3. \tag{6.2}$$

Array 4. *The other factors are respectively equal to:*

Cost of type 1': $\frac{\mu}{2}\left[1 - \frac{\mu}{2}(j-1)\right] \; \mathbf{P}_s \circ \Delta\mathbf{H}_s \circ \mathbf{P}_s \circ \mathbf{H}_s^{[b]} \circ \mathbf{P}_s[f](0)$

Cost of type 2: $\frac{1}{2}\left(\frac{\mu}{2}\right)^2 \; \mathbf{P}_s \circ \mathbf{H}_s^{[c]} \circ \mathbf{P}_s \circ \Delta\mathbf{H}_s \circ \mathbf{P}_s[f](0)$

Cost of type 3: $4\frac{\mu}{2}\left[1 - \frac{\mu}{2}(j-1)\right] \; \mathbf{P}_s \circ \Delta\mathbf{H}_s \circ \mathbf{P}_s \circ \Delta\mathbf{H}_s \circ \mathbf{P}_s[f](0)$

Near $s = 2$, all these functions are analytic and their values at $s = 2$ involve the same integrals as in (5.7). As in Section 5.4, it is easy to compare costs with a hat and costs with a bar. This ends the proof of the Lemma ∎

When summing over index j that varies between 1 and $\bar{J} := \lceil 2/\mu \rceil$, one gets the expression given in Theorem 3. It then remains to compare the bit-complexity of the two algorithms, the Lehmer–Euclid algorithm $\mathcal{LE}_\mu$ and its regularized version $\overline{\mathcal{LE}}_\mu$. This is the purpose of the following Lemma.

Lemma 11. *When $n \to \infty$, the average bit–complexity of the Lehmer–Euclid algorithm $\mathcal{LE}_\mu$ and the average bit–complexity of its regularized version $\overline{\mathcal{LE}}_\mu$ are asymptotically the same.*

Proof. Lemma 9 proves that the length $\delta(j)$ of the j-th phase is almost surely close to $\bar{p} := \lfloor (\mu/2)p \rfloor$. We then split the set of inputs Ω into two subsets, an exceptional subset, and an ordinary subset. Consider, for some $\varepsilon > 0$ the event

$$D(\varepsilon) := [\exists j \leq J, \quad |\delta(j) - \bar{p}| > \varepsilon p]. \tag{6.3}$$

Lemma 9 proves that there exists $K < 1$ for which $\Pr_n[D(\varepsilon)] = O(K^n)$, so that, the subset $D(\varepsilon)$ is exceptional. It is sufficient to study the bit-complexity of the $\mathcal{LE}_\mu$ algorithm on the complementary subset of $D(\varepsilon)$. Here, there are two "extremal ordinary values"

$$\bar{p}_- := \left\lfloor \left(\frac{\mu}{2} - \varepsilon\right) p \right\rfloor, \qquad \bar{p}_+ := \left\lceil \left(\frac{\mu}{2} + \varepsilon\right) p \right\rceil,$$

and the beginning indices $p(j), p(j+1)$ satisfy

$$j\bar{p}_- \leq p(j) \leq j\bar{p}_+,$$
$$[j\bar{p}_+, (j+1)\bar{p}_-] \subset [p(j), p(j+1)] \subset [j\bar{p}_-, (j+1)\bar{p}_+].$$

Note that the number J of phases satisfies

$$J^- := \frac{2}{\mu + 2\varepsilon} \leq J \leq J^+ := \frac{2}{\mu - 2\varepsilon} \tag{6.4}$$

and remark that the length of the large interval $[j\bar{p}_-, (j+1)\bar{p}_+]$ is less than $\bar{p}_{++} := \bar{p} + J^+\varepsilon p$, while the length of the small interval is more than $\bar{p}_{--} := \bar{p} - J^-\varepsilon p$. The function $i \to \ell(A_i)$ is a decreasing function, and the function $[r,t] \to v[r,t]$ is an increasing function. Finally, for each cost that has been previously defined, we provide an upper bound and a lower bound on the ordinary subset.

Array 5. *Upper ordinary bounds.*

Cost of type 1':	$\sum_{j=1}^{J^+} \ell(A_{(j-1)\bar{p}_-}) \cdot \sum_{i=(j-1)\bar{p}_-}^{j\bar{p}_+ - 1} b(q_i)$
Cost of type 2:	$2\sum_{j=1}^{J^+} \sum_{i=(j-1)\bar{p}_-}^{j\bar{p}_+ - 1} \ell(v[(j-1)\bar{p}_-, i]) \cdot c(q_i)$
Cost of type 3:	$4\sum_{j=1}^{J^+} \ell(A_{(j-1)\bar{p}_-}) \cdot \ell(v[(j-1)\bar{p}_-, j\bar{p}_+])$

Array 6. *Lower ordinary bounds.*

Cost of type 1':	$\sum_{j=1}^{J^-} \ell(A_{(j-1)\bar{p}_+}) \cdot \sum_{i=(j-1)\bar{p}_+}^{j\bar{p}_- - 1} b(q_i)$
Cost of type 2:	$2\sum_{j=1}^{J^-} \sum_{i=(j-1)\bar{p}_+}^{j\bar{p}_- - 1} \ell(v[(j-1)\bar{p}_+, i]) \cdot c(q_i)$
Cost of type 3:	$4\sum_{j=1}^{J^-} \ell(A_{(j-1)\bar{p}_+}) \cdot \ell(v[(j-1)\bar{p}_+, j\bar{p}_-])$

Here are the Dirichlet series of various costs relative to j-th phase, when the total depth of the algorithm equals p. In fact, we work with some approximate costs that are the analogs of the costs with a hat of the previous Section.

Array 7. *Dirichlet series for upper ordinary bounds of costs during the j-th phase.*

Cost of type 1':	$\sum_{i=1}^{\bar{p}++} \Delta \left[\mathbf{H}_s^{p-(j-1)\bar{p}--i} \circ \mathbf{H}_s^{[b]} \circ \mathbf{H}_s^{i-1} \right] \circ \mathbf{H}_s^{(j-1)\bar{p}-}[f](0)$
Cost of type 2:	$2\sum_{i=1}^{\bar{p}++} \mathbf{H}_s^{p-(j-1)\bar{p}--i} \circ \mathbf{H}_s^{[c]} \circ \Delta\left[\mathbf{H}_s^{i-1}\right] \circ \mathbf{H}_s^{(j-1)\bar{p}-}[f](0)$
Cost of type 3:	$4\Delta \left[\mathbf{H}_s^{p-j\bar{p}+} \circ \Delta\left(\mathbf{H}_s^{\bar{p}++}\right)\right] \circ \mathbf{H}_s^{(j-1)\bar{p}-}[f](0)$

Array 8. *Dirichlet series for lower ordinary bounds of costs during the j-th phase.*

Cost of type 1':	$\sum_{i=1}^{\bar{p}--} \Delta \left[\mathbf{H}_s^{p-(j-1)\bar{p}+-i} \circ \mathbf{H}_s^{[b]} \circ \mathbf{H}_s^{i-1} \right] \circ \mathbf{H}_s^{(j-1)\bar{p}+}[f](0)$
Cost of type 2:	$2\sum_{i=1}^{\bar{p}--} \mathbf{H}_s^{p-(j-1)\bar{p}+-i} \circ \mathbf{H}_s^{[c]} \circ \Delta\left[\mathbf{H}_s^{i-1}\right] \circ \mathbf{H}_s^{(j-1)\bar{p}+}[f](0)$
Cost of type 3:	$4\Delta \left[\mathbf{H}_s^{p-j\bar{p}-} \circ \Delta\left(\mathbf{H}_s^{\bar{p}--}\right)\right] \circ \mathbf{H}_s^{(j-1)\bar{p}+}[f](0)$

When summing over all possible values of depth p, the dominant terms of the Dirichlet series of Array 7 or Array 8 all involve a product of two factors. The first factor is the same for all the three series of the two arrays and is the same as in (6.2), i.e.,

$$\sum_{p\geq 0} p^2 \lambda(s)^p \sim 2\left(\frac{1}{1-\lambda(s)}\right)^3 .$$

It remains to study the second factors and compare them to the expressions given in Array 4. The operators involved are the same as in Array 4, and the constants of Array 4

$$A(\mu) := \frac{\mu}{2}\left[1 - \frac{\mu}{2}(j-1)\right], \qquad B(\mu) := \frac{1}{2}\left(\frac{\mu}{2}\right)^2$$

are respectively replaced by $A(\mu)+O(\varepsilon), B(\mu)+O(\varepsilon)$. We now sum over all possible values of index j. This index varies (according to the cases) between 1 and J^- or between 1 and J^+ (these values are defined in (6.4)). We finally obtain the result. ∎

This ends the proof of Theorem 3 which is the main result of the paper.

The previous results directly show that all happens as if we could apply directly the heuristic reasoning of Section 2. We recall that Section 3 provides a first explanation of this fact, based on the description of the evolution of the density on Ω during the execution of the Euclidean Algorithm.

7. Conclusion.

This paper provides the first average–case analysis of the Lehmer–Euclid algorithm when the truncation degree μ is constant. However, this is not this algorithm that is used in the real life. Usually, one uses truncations that transform multi–precision integers into single-precision integers, so that the truncation degree is no more constant, since it is of the form c/n with some constant c. If we wish dealing with this "real life" algorithm, we have to change our analysis which only works with constant values of μ. In this case, we need some uniform results for which remainder terms are essential, and it is not possible to obtain such terms with Tauberian Theorems. We hope that some new results [2] about the Euclidean transfer operator may be perhaps used for obtaining such remainder terms. Another way to deal with the $\mathcal{LE}$ algorithm is to directly analyse another interrupted Euclidean algorithm that stops when the current integer has lost a constant number of bits.

However, this first study may be quite interesting from other points of view. Here, we have analyzed a standard version of the parameterized Lehmer-Euclid Algorithm. Collins [7], Jebelean [15], and many others proposed variant improvements on the Lehmer-Euclid algorithm; for instance, it is sufficient to compute only one co-sequence, and recover the second one at the end of each phase, with a supplementary product. On the otherside, Jebelean remarks that, when using truncated numbers with m bits, almost all operations deal with numbers of $m/2$ bits, and very few operations actually use numbers with m-bits. Then, he proposes to work with double-precision numbers. Until now, the "measure" of these various improvements is only experimental; here, we provide a new tool that can be used as an alternative measure of these improvements.

Finally, there exists another version of the Lehmer-Euclid Algorithm, a recursive one, that we denote as $\mathcal{REL}$. It is based on a "Divide and Conquer" principle, and replaces computations on n–bit integers by operations on μn–bit integers (with μ near $1/2$). The design of this algorithm, initially proposed by Schonhage [21], is now clear, after recent progress due to the works of Cesari [6], Zimmermann [29] and Stehlé [23]. It is now well-described in the book of Yap [28]. It is our further project to analyse the $\mathcal{REL}$ Algorithm and we think that many tools that we have introduced here can be used in this further work.

Acknowledgements. We wish to thank Philippe Flajolet and Damien Stelhé for many interesting discussions related to this paper.

References

[1] Akhavi, A., Vallée, B. Average bit–complexity of Euclidean Algorithms, Proceedings of ICALP'00, Lecture Notes in Computer Science 1853, pp 373-387, Springer.

[2] Baladi, V., Vallée, B. Dynamical analysis "in distribution", in preparation.

[3] Bedford, T., Keane, M., and Series, C., Eds. *Ergodic Theory, Symbolic Dynamics and Hyperbolic Spaces*, Oxford University Press, 1991.

[4] Brent, R.P. Analysis of the Binary Euclidean algorithm, Algorithms and Complexity, New directions and recent results, ed. by J.F. Traub, Academic Press 1976, pp 321–355

[5] Bourdon, J. Daireaux, B. Vallée, B. Dynamical analysis of α-Euclidean Algorithms, Journal of Algorithms 44 (2002) pp 246–285.
[6] Cesari, G. Parallel Implementation of Schonhage's Integer GCD Algorithm, Proceedings of ANTS-III, LNCS 1423, pp64-76.
[7] Collins, G.E. Lecture Notes on arithmetic algorithms, Univ. of Wisconsin.
[8] Delange, H. Généralisation du Théorème d'Ikehara, *Ann. Sc. ENS, (1954) 71, pp 213–242.*
[9] Dixon, J. D. The number of steps in the Euclidean algorithm, *Journal of Number Theory 2* (1970), 414–422.
[10] Flajolet, P. Analytic analysis of algorithms, In Proceedings of the 19th International Colloquium "Automata, Languages and Programming", Vienna, July 1992, W. Kuich, editor, Lecture Notes in Computer Science 623, pp 186–210
[11] Flajolet, P. and Sedgewick, R. Analytic Combinatorics, Book in preparation (1999), see also INRIA Research Reports 1888, 2026, 2376, 2956.
[12] Heilbronn, H. On the average length of a class of continued fractions, Number Theory and Analysis, ed. by P. Turan, New-York, Plenum, 1969, pp 87-96.
[13] Hensley, D. The number of steps in the Euclidean algorithm, *Journal of Number Theory* 49, 2 (1994), 142–182.
[14] Jebelean, T. Practical Integer Division with Karatsuba Complexity Proceedings of ISSAC'97
[15] Jebelean, T. A Double-Digit Lehmer–Euclid Algorithm for finding the GCD of Long Integers. Journal of Symbolic Computation (1995) 19, pp 145–157
[16] Krandick, W., and Johnson, J.R. Efficient multi-precision floating point multiplication with optimal directional rounding, Proceedings of the 11th IEEE Symposium in Computer Arithmetic, pp 228–233.
[17] Knuth, D.E., The art of Computer programming, Volume 2, 3rd edition, Addison Wesley, Reading, Massachussets, 1998.
[18] Lehmer, D. H. Euclid's algorithm for large numbers. Am. Math. Mon. (1938) 45 pp 227–233.
[19] Lercier, R. Algorithmique des courbes elliptiques dans les corps finis, PhD Thesis, Ecole Polytechnique (1997).
[20] Ruelle, D. *Thermodynamic formalism,* Addison Wesley (1978)
[21] Schonhage, A. Schnelle Berechnung von Kettenbruchentwicklungen, Acta Informatica pp 139–144 (1971)
[22] Sorenson, J. An analysis of Lehmer's Euclidean GCD Algorithm, Proceedings of ISSAC 1995, pp 254–258
[23] Stehlé, D. Master's Thesis, University of Caen (2001)
[24] Tenenbaum, G. *Introduction à la théorie analytique des nombres*, vol. 13. Institut Élie Cartan, Nancy, France, 1990.
[25] Vallée, B. Dynamics of the Binary Euclidean Algorithm: Functional Analysis and Operators., Algorithmica (1998) vol 22 (4) pp 660–685.
[26] Vallée, B. Dynamical Analysis of a Class of Euclidean Algorithms, to appear in Theoretical Computer Science (2003)
[27] Vallée, B. Digits and Continuants in Euclidean Algorithms. Ergodic Versus Tauberian Theorems, Journal de Théorie des Nombres de Bordeaux 12 (2000) pp 531-570.
[28] Yap, C.K. Fundamental Problems in Algorithmic Algebra, Princeton University Press (1996)
[29] Zimmermann, P. A proof of subquadratic gcd, preprint (2001)

GREYC, Université de Caen, F-14032 Caen (France).