An instance of analysis of algorithms: The plain GCD algorithm

Brigitte Vallée,
Laboratoire GREYC,
CNRS and University of Caen, France

An instance of analysis of algorithms: The plain GCD algorithm

Brigitte Vallée,
Laboratoire GREYC,
CNRS and University of Caen, France

Based on a joint work
with Valérie Berthé, Jean Creusefond and Loïck Lhote.

Journées nationales du GDR Informatique Mathématique
Bordeaux, Février 2015.

Computing the gcd of ℓ inputs

For $\ell = 2$: the "classical" Euclid algorithm :

a sequence of Euclidean divisions

For $\ell \geq 3$, there are various strategies.

Computing the gcd of ℓ inputs

For $\ell = 2$: the "classical" Euclid algorithm :

a sequence of Euclidean divisions

For $\ell \geq 3$, there are various strategies.

The plain algorithm performs a sequence of computations on two entries;

On the input $(x_1, x_2, \ldots, x_\ell)$, it computes

– first: $y_2 := \gcd(x_1, x_2)$

– then, for $k \in [3..\ell]$: $y_k := \gcd(x_k, y_{k-1}) = \gcd(x_1, x_2, \ldots, x_k)$.

Computing the gcd of ℓ inputs

For $\ell = 2$: the "classical" Euclid algorithm :

a sequence of Euclidean divisions

For $\ell \geq 3$, there are various strategies.

The plain algorithm performs a sequence of computations on two entries;

On the input $(x_1, x_2, \ldots, x_\ell)$, it computes

– first: $y_2 := \gcd(x_1, x_2)$

– then, for $k \in [3..\ell]$: $y_k := \gcd(x_k, y_{k-1}) = \gcd(x_1, x_2, \ldots, x_k)$.

The "total" gcd $y_\ell := \gcd(x_1, x_2, \ldots, x_\ell)$ is obtained after $\ell - 1$ phases.

Each phase performs a call to the classical Euclid algorithm.

Computing the gcd of ℓ inputs

For $\ell = 2$: the "classical" Euclid algorithm :

a sequence of Euclidean divisions

For $\ell \geq 3$, there are various strategies.

The plain algorithm performs a sequence of computations on two entries;

On the input $(x_1, x_2, \ldots, x_\ell)$, it computes

– first: $y_2 := \gcd(x_1, x_2)$

– then, for $k \in [3..\ell]$: $y_k := \gcd(x_k, y_{k-1}) = \gcd(x_1, x_2, \ldots, x_k)$.

The "total" gcd $y_\ell := \gcd(x_1, x_2, \ldots, x_\ell)$ is obtained after $\ell - 1$ phases.

Each phase performs a call to the classical Euclid algorithm.

The same formal scheme

– for polynomials over a finite field: $\mathbb{F}_q[X]$

– for numbers : positive integers.

Computing the gcd of ℓ inputs

For $\ell = 2$: the "classical" Euclid algorithm :

a sequence of Euclidean divisions

For $\ell \geq 3$, there are various strategies.

The plain algorithm performs a sequence of computations on two entries;

On the input $(x_1, x_2, \ldots, x_\ell)$, it computes

– first: $y_2 := \gcd(x_1, x_2)$

– then, for $k \in [3..\ell]$: $y_k := \gcd(x_k, y_{k-1}) = \gcd(x_1, x_2, \ldots, x_k)$.

The "total" gcd $y_\ell := \gcd(x_1, x_2, \ldots, x_\ell)$ is obtained after $\ell - 1$ phases.

Each phase performs a call to the classical Euclid algorithm.

The same formal scheme

– for polynomials over a finite field: $\mathbb{F}_q[X]$

– for numbers : positive integers.

A very natural scheme, proposed in Knuth's book.....

Computing the gcd of ℓ inputs

For $\ell = 2$: the "classical" Euclid algorithm :

a sequence of Euclidean divisions

For $\ell \geq 3$, there are various strategies.

The plain algorithm performs a sequence of computations on two entries;

On the input $(x_1, x_2, \ldots, x_\ell)$, it computes

– first: $y_2 := \gcd(x_1, x_2)$

– then, for $k \in [3..\ell]$: $y_k := \gcd(x_k, y_{k-1}) = \gcd(x_1, x_2, \ldots, x_k)$.

The "total" gcd $y_\ell := \gcd(x_1, x_2, \ldots, x_\ell)$ is obtained after $\ell - 1$ phases.

Each phase performs a call to the classical Euclid algorithm.

The same formal scheme

– for polynomials over a finite field: $\mathbb{F}_q[X]$

– for numbers : positive integers.

A very natural scheme, proposed in Knuth's book.....

but not yet analyzed for $\ell > 2$ (Problem HM 48)

Which behavior can be expected?

Knuth wrote: "In most cases, the size of the partial gcd decreases rapidly during the first few phases of the calculation.
This will make the remainder of the computation quite fast".

Which behavior can be expected?

Knuth wrote: "In most cases, the size of the partial gcd decreases rapidly during the first few phases of the calculation.
This will make the remainder of the computation quite fast".

Our analysis exhibits a more precise phenomenon:
A strong difference between the first phase and the subsequent phases.
In most cases, "almost all the calculation" is done during the first phase.

Which behavior can be expected?

Knuth wrote: "In most cases, the size of the partial gcd decreases rapidly during the first few phases of the calculation.
This will make the remainder of the computation quite fast".

Our analysis exhibits a more precise phenomenon:
A strong difference between the first phase and the subsequent phases.
In most cases, "almost all the calculation" is done during the first phase.

We prove the following facts about the number of divisions performed, measured with respect to the size of the input:

- during the first phase:
  - it is linear on average,
  - it asymptotically follows a beta law;

Which behavior can be expected?

Knuth wrote: "In most cases, the size of the partial gcd decreases rapidly during the first few phases of the calculation.
This will make the remainder of the computation quite fast".

Our analysis exhibits a more precise phenomenon:
A strong difference between the first phase and the subsequent phases.
In most cases, "almost all the calculation" is done during the first phase.

We prove the following facts about the number of divisions performed, measured with respect to the size of the input:

- during the first phase:
  - it is linear on average,
  - it asymptotically follows a beta law;
- during subsequent phases:
  - it is constant on average
  - it asymptotically follows a geometric law

Which behavior can be expected?

Knuth wrote: "In most cases, the size of the partial gcd decreases rapidly during the first few phases of the calculation.
This will make the remainder of the computation quite fast".

Our analysis exhibits a more precise phenomenon:
A strong difference between the first phase and the subsequent phases.
In most cases, "almost all the calculation" is done during the first phase.

We prove the following facts about the number of divisions performed, measured with respect to the size of the input:

- during the first phase:
  - it is linear on average,
  - it asymptotically follows a beta law;
- during subsequent phases:
  - it is constant on average
  - it asymptotically follows a geometric law

The same phenomena occur for the size of the partial gcd.

Plan of the talk.

Plan of the talk.

I. Probabilistic analysis of algorithms.

Based on Analytic Combinatorics and Generating Functions.

Plan of the talk.

I. Probabilistic analysis of algorithms.

Based on Analytic Combinatorics and Generating Functions.

II. Analysis of the polynomial case.

Based on Analytic Combinatorics and power Generating Functions.

Plan of the talk.

I. Probabilistic analysis of algorithms.

Based on Analytic Combinatorics and Generating Functions.

II. Analysis of the polynomial case.

Based on Analytic Combinatorics and power Generating Functions.

III. Analysis of the integer case.

Based on Dynamical Combinatorics and Dirichlet Generating Functions

Plan of the talk.

I. Probabilistic analysis of algorithms.

Based on Analytic Combinatorics and Generating Functions.

II. Analysis of the polynomial case.

Based on Analytic Combinatorics and power Generating Functions.

III. Analysis of the integer case.

Based on Dynamical Combinatorics and Dirichlet Generating Functions

IV. An unified point of view for the two analyses?

A dynamical point of view.

I. Probabilistic analysis of algorithms

Based on Analytic Combinatorics
and Generating Functions

Probabilistic analysis of an algorithm

– The set of the possible inputs for the algorithm is denoted by Ω.

Probabilistic analysis of an algorithm

– The set of the possible inputs for the algorithm is denoted by Ω.

– There is a size function $\mathrm{d} : \Omega \to \mathbb{N}$.

  – The subset $\Omega_n := \{\omega : \mathrm{d}(\omega) = n\}$ is finite

  – It is endowed with the uniform distribution $\mathbb{P}_n$.

Probabilistic analysis of an algorithm

- The set of the possible inputs for the algorithm is denoted by Ω.
- There is a size function $\mathrm{d} : \Omega \to \mathbb{N}$.
  - The subset $\Omega_n := \{\omega : \mathrm{d}(\omega) = n\}$ is finite
  - It is endowed with the uniform distribution $\mathbb{P}_n$.
- There is a cost function $L : \Omega \to \mathbb{N}$,
  - study the probabilistic behavior of L on each Ω_n
    - estimate its mean, its variance, its distribution,
  - in an asymptotic way (for $n \to \infty$).

Probabilistic analysis of an algorithm

– The set of the possible inputs for the algorithm is denoted by Ω.

– There is a size function $\mathrm{d} : \Omega \to \mathbb{N}$.

  – The subset $\Omega_n := \{\omega : \mathrm{d}(\omega) = n\}$ is finite

  – It is endowed with the uniform distribution $\mathbb{P}_n$.

– There is a cost function $L : \Omega \to \mathbb{N}$,

  – study the probabilistic behavior of L on each Ω_n

    – estimate its mean, its variance, its distribution,

  – in an asymptotic way (for $n \to \infty$).

$$\mathbb{E}_n[L] \underset{n\to\infty}{\sim} a_n, \quad \mathbb{V}_n[L] \underset{n\to\infty}{\sim} b_n, \quad \mathbb{P}_n\left[\frac{L - a_n}{\sqrt{b_n}} \in [x, x + dx]\right] \underset{n\to\infty}{\sim} f(x)dx$$

Examples of limit laws (discrete or continuous)

x-axis: possible values of the cost $L(\omega)$

y-axis: probability density $x \mapsto f(x)$

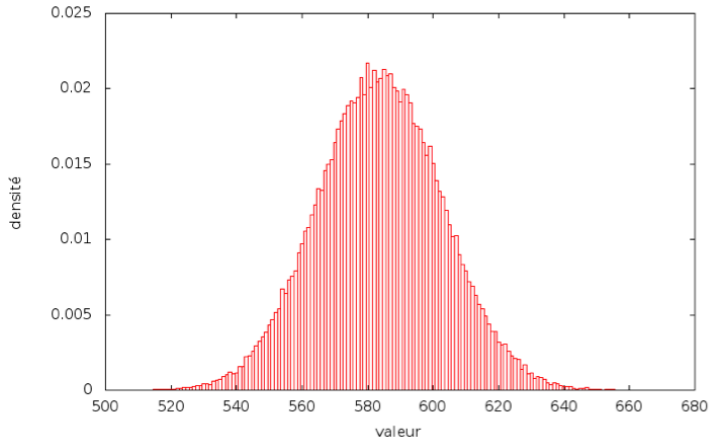
$f(x)dx := \mathbb{P}[\omega; \quad L(\omega) \in [x, x + dx]]$

Examples of limit laws (discrete or continuous)

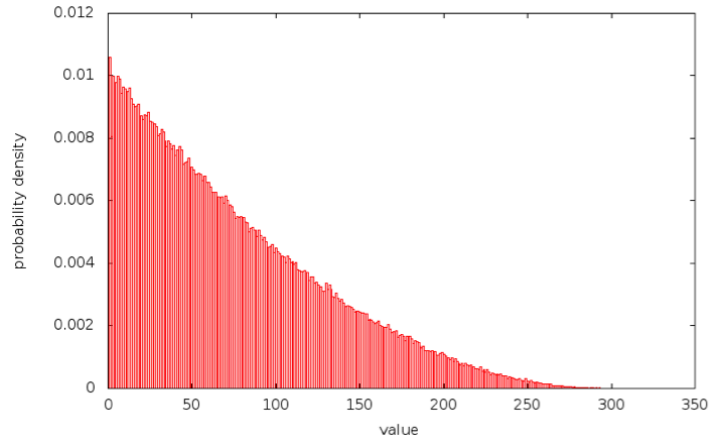
x-axis: possible values of the cost $L(\omega)$

y-axis: probability density $x \mapsto f(x)$

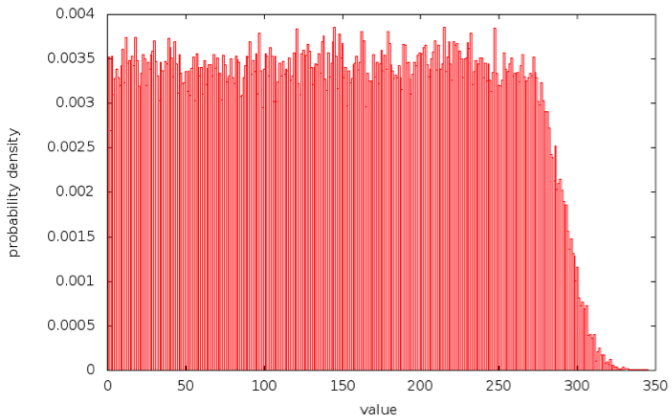
$f(x)dx := \mathbb{P}[\omega; \quad L(\omega) \in [x, x+dx]]$



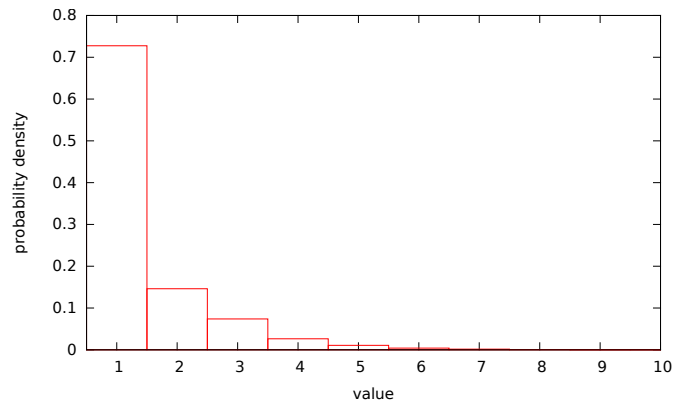
Gaussian law $f(t) \asymp e^{-t^2/2}$



Beta law (a, b) $f(t) \asymp t^{a-1}(1-t)^{b-1}$



Uniform law $f(t) \asymp 1$



A discrete law: Geometric law $f(n) \asymp a^n$

Usual analysis for the Euclid Algorithm $\ell = 2$.

The usual size of $\underline{x} = (x_1, x_2)$ is the maximum of the sizes of the inputs:

$$\mathrm{d}(\underline{x}) := \max(\mathrm{d}(x_1), \mathrm{d}(x_2)).$$

The two costs of interest are :

– the number L of steps – the size D of the output gcd.

Usual analysis for the Euclid Algorithm $\ell = 2$.

The usual size of $\underline{x} = (x_1, x_2)$ is the maximum of the sizes of the inputs:

$$\mathrm{d}(\underline{x}) := \max(\mathrm{d}(x_1), \mathrm{d}(x_2)).$$

The two costs of interest are :

– the number L of steps – the size D of the output gcd.

Main results:

Usual analysis for the Euclid Algorithm $\ell = 2$.

The usual size of $\underline{x} = (x_1, x_2)$ is the maximum of the sizes of the inputs:

$$\mathrm{d}(\underline{x}) := \max(\mathrm{d}(x_1), \mathrm{d}(x_2)).$$

The two costs of interest are :

– the number L of steps – the size D of the output gcd.

Main results:

The number L of divisions

has a mean value $\mathbb{E}_n[L]$ of linear order;

Its distribution is asymptotically gaussian.

Usual analysis for the Euclid Algorithm $\ell = 2$.

The usual size of $\underline{x} = (x_1, x_2)$ is the maximum of the sizes of the inputs:

$$\mathrm{d}(\underline{x}) := \max(\mathrm{d}(x_1), \mathrm{d}(x_2)).$$

The two costs of interest are :

– the number L of steps – the size D of the output gcd.

Main results:

The number L of divisions

has a mean value $\mathbb{E}_n[L]$ of linear order;

Its distribution is asymptotically gaussian.

The size D of the gcd

has a mean value $\mathbb{E}_n[D]$ of constant order;

Its distribution is asymptotically geometric.

Usual analysis for the Euclid Algorithm $\ell = 2$.

The usual size of $\underline{x} = (x_1, x_2)$ is the maximum of the sizes of the inputs:

$$\mathrm{d}(\underline{x}) := \max(\mathrm{d}(x_1), \mathrm{d}(x_2)).$$

The two costs of interest are :

– the number L of steps – the size D of the output gcd.

Main results:

The number L of divisions

has a mean value $\mathbb{E}_n[L]$ of linear order;

Its distribution is asymptotically gaussian.

The size D of the gcd

has a mean value $\mathbb{E}_n[D]$ of constant order;

Its distribution is asymptotically geometric.

This (usual) size is not adapted for a number of inputs $\ell > 2$.

For $\ell \geq 2$, we consider the total size and this choice modifies the results

$\Longrightarrow$ Irruption of the (unexpected) beta law

Analysis of algorithms and generating functions.

Two main cases here, depending on

– the type of the input $\underline{x} = (x_1, x_2, \ldots, x_\ell)$, $x_i \in \mathbb{N}$ or $x_i \in \mathbb{F}_q[X]$

– the natural size on the input : the size of $\prod x_i$.

Analysis of algorithms and generating functions.

Two main cases here, depending on

– the type of the input $\underline{x} = (x_1, x_2, \ldots, x_\ell)$, $x_i \in \mathbb{N}$ or $x_i \in \mathbb{F}_q[X]$

– the natural size on the input : the size of $\prod x_i$.

– for polynomials: the total degree $\mathrm{d}(\underline{x}) = \mathrm{d}(\prod x_i) = \sum \mathrm{d}(x_i)$

$\Omega_n := \{\underline{x} = (x_1, x_2, \ldots, x_\ell); \quad \mathrm{d}(\prod x_i) = n\}$

Power generating functions for polynomials

$$S(z) := \sum_{\underline{x} \in \Omega} z^{\mathrm{d}(\prod x_i)} \quad \Longrightarrow \quad |\Omega_n| = [z^n] S(z)$$

Analysis of algorithms and generating functions.

Two main cases here, depending on

– the type of the input $\underline{x} = (x_1, x_2, \ldots, x_\ell)$, $x_i \in \mathbb{N}$ or $x_i \in \mathbb{F}_q[X]$

– the natural size on the input : the size of $\prod x_i$.

– for polynomials: the total degree $\quad \mathrm{d}(\underline{x}) = \mathrm{d}(\prod x_i) = \sum \mathrm{d}(x_i)$

$$\Omega_n := \{\underline{x} = (x_1, x_2, \ldots, x_\ell); \quad \mathrm{d}(\prod x_i) = n\}$$

Power generating functions for polynomials

$$S(z) := \sum_{\underline{x} \in \Omega} z^{\mathrm{d}(\prod x_i)} \quad \Longrightarrow \quad |\Omega_n| = [z^n] S(z)$$

– for integers: the total length $\quad \mathrm{d}(\underline{x}) = \lfloor \log \prod x_i \rfloor \sim \sum \lfloor \log x_i \rfloor$

$$\Omega_n := \{\underline{x} = (x_1, x_2, \ldots, x_\ell); \quad e^n \leq \prod x_i < e^{n+1}\}$$

Dirichlet generating functions for integers

$$S(s) := \sum_{\underline{x} \in \Omega} (\prod x_i)^{-s} \quad \Longrightarrow \quad |\Omega_n| = \sum_{e^n \leq N < e^{n+1}} [N^{-s}] S(s)$$

Analysis of algorithms and generating functions (II).

$$\mathbb{P}_n[L=m]=\frac{|\Omega_{n,m}|}{|\Omega_n|} \qquad \text{with} \quad \Omega_{n,m}=\{\underline{x}\in\Omega_n;\quad L(\underline{x})=m\}$$

Numerators are expressed with coefficients of bivariate generating functions.

For polynomials:

$$\Omega_{n,m}:=\{\underline{x}=(x_1,x_2,\ldots,x_\ell\};\quad \mathrm{d}(\textstyle\prod x_i)=n,\quad L(\underline{x})=m\}$$

$$S(z):=\sum_{\underline{x}\in\Omega} z^{\mathrm{d}(\prod x_i)}, \qquad S(z,u):=\sum_{\underline{x}\in\Omega} z^{\mathrm{d}(\prod x_i)}u^{L(\underline{x})}$$

$$\mathbb{P}_n[L=m]=\frac{[z^nu^m]S(z,u)}{[z^n]S(z)}$$

For integers:

$$\Omega_{n,m}:=\{\underline{x}=(x_1,x_2,\ldots,x_\ell);\quad e^n\le\textstyle\prod x_i<e^{n+1},\quad L(\underline{x})=m\}$$

$$S(s):=\sum_{\underline{x}\in\Omega}(\textstyle\prod x_i)^{-s}, \qquad S(s,u):=\sum_{\underline{x}\in\Omega}(\textstyle\prod x_i)^{-s}u^{L(\underline{x})},$$

$$\mathbb{P}_n[L=m]=\frac{\displaystyle\sum_{e^n\le N<e^{n+1}}[N^{-s}u^m]S(s,u)}{\displaystyle\sum_{e^n\le N<e^{n+1}}[N^{-s}]S(s)}$$

Analysis of algorithms and generating functions (GF): main principles.

Distributional analysis is related

– to the asymptotic behaviour of coefficients of the BGF's

– obtained with two steps:

Combinatorial step.

Translate the structure of the algorithm on the generating functions:

- obtain an algorithmic expression of the generating function,
- from which the dominant singularity becomes apparent.

Analytic step.

For any GF, the asymptotic behaviour of coefficients is related to

- the position and the nature of its dominant singularity
- when the GF is viewed as a function of the complex variable.

Analysis of algorithms and generating functions (GF): main principles.

Distributional analysis is related

– to the asymptotic behaviour of coefficients of the BGF's

– obtained with two steps:

Combinatorial step.

Translate the structure of the algorithm on the generating functions:

- obtain an algorithmic expression of the generating function,
- from which the dominant singularity becomes apparent.

Analytic step.

For any GF, the asymptotic behaviour of coefficients is related to

- the position and the nature of its dominant singularity
- when the GF is viewed as a function of the complex variable.

The same principles in the two analyses and two main differences.

– Dirichlet GF's more difficult to study than power GF's

– For integers, there are correlations due to carries.

II. Analysis of the polynomial case.

Based on Analytic Combinatorics
and Generating Functions

Probabilistic analysis of the plain ℓ–GCD algorithm on $\mathbb{F}_q[X]$.

On the input $(x_1, x_2, \ldots, x_\ell)$,

- the algorithm computes the total gcd $y_\ell := \gcd(x_1, x_2, \ldots, x_\ell)$
- with $\ell - 1$ phases.
- The k-th phase computes the k–th gcd,

$$y_k := \gcd(x_k, y_{k-1}) = \gcd(x_1, x_2, \ldots, x_k) \ .$$

Probabilistic analysis of the plain ℓ–GCD algorithm on $\mathbb{F}_q[X]$.

On the input $(x_1, x_2, \ldots, x_\ell)$,

- the algorithm computes the total gcd $y_\ell := \gcd(x_1, x_2, \ldots, x_\ell)$
- with $\ell - 1$ phases.
- The k-th phase computes the k–th gcd,

$$y_k := \gcd(x_k, y_{k-1}) = \gcd(x_1, x_2, \ldots, x_k) \ .$$

- each phase performs the classical Euclid algorithm

via a sequence of Euclidean divisions

Probabilistic analysis of the plain ℓ–GCD algorithm on $\mathbb{F}_q[X]$.

On the input $(x_1, x_2, \ldots, x_\ell)$,

- the algorithm computes the total gcd $y_\ell := \gcd(x_1, x_2, \ldots, x_\ell)$
- with $\ell - 1$ phases.
- The k-th phase computes the k–th gcd,

$$y_k := \gcd(x_k, y_{k-1}) = \gcd(x_1, x_2, \ldots, x_k) \ .$$

- each phase performs the classical Euclid algorithm

via a sequence of Euclidean divisions

The set of inputs is $\Omega = \{(x_1, \ldots, x_\ell); \quad x_i \ \text{ monic } \ \in \mathbb{F}_q[X]\}$

The size of an input : $\mathrm{d}(x_1, \ldots, x_\ell) = \mathrm{d}(x_1) + \ldots + \mathrm{d}(x_\ell)$,

with $\mathrm{d}(x) := \deg(x)$.

Probabilistic analysis of the plain ℓ–GCD algorithm on $\mathbb{F}_q[X]$.

On the input $(x_1, x_2, \ldots, x_\ell)$,

- the algorithm computes the total gcd $y_\ell := \gcd(x_1, x_2, \ldots, x_\ell)$
- with $\ell - 1$ phases.
- The k-th phase computes the k–th gcd,

$$y_k := \gcd(x_k, y_{k-1}) = \gcd(x_1, x_2, \ldots, x_k) \ .$$

- each phase performs the classical Euclid algorithm

via a sequence of Euclidean divisions

The set of inputs is $\Omega = \{(x_1, \ldots, x_\ell); \quad x_i \ \text{ monic } \ \in \mathbb{F}_q[X]\}$

The size of an input : $\mathrm{d}(x_1, \ldots, x_\ell) = \mathrm{d}(x_1) + \ldots + \mathrm{d}(x_\ell)$,

with $\mathrm{d}(x) := \deg(x)$.

Main costs of interest

- the number L_k of divisions during the k–th phase

i.e. on the input (x_k, y_{k-1})

- the degree D_k of the k–th gcd

(at the beginning of the k-th phase).

The combinatorial bijection induced by the Euclid Algorithm $[\ell = 2]$

The combinatorial bijection induced by the Euclid Algorithm [$\ell = 2$]

`Euclid`(a_1, a_2), [case $\mathrm{d}(a_1) \geq \mathrm{d}(a_2)$].

$$
\begin{array}{lcllll}
a_1 & = & m_1\, a_2 & + & a_3 & 0 < \mathrm{d}(a_3) < \mathrm{d}(a_2) \\
a_2 & = & m_2\, a_3 & + & a_4 & 0 < \mathrm{d}(a_4) < \mathrm{d}(a_3) \\
\ldots & = & \ldots & + & & \\
a_{r-1} & = & m_{r-1}\, a_r & + & a_{r+1} & 0 < \mathrm{d}(a_{r+1}) < \mathrm{d}(a_r) \\
a_r & = & m_r\, a_{r+1} & + & 0 &
\end{array}
$$

The last non zero remainder is the gcd y. Here $y = a_{r+1}$.

The combinatorial bijection induced by the Euclid Algorithm [$\ell = 2$]

$\texttt{Euclid}(a_1, a_2)$, [case $\mathrm{d}(a_1) \geq \mathrm{d}(a_2)$].

$$
\begin{array}{lcllll}
a_1 & = & m_1\, a_2 & + & a_3 & 0 < \mathrm{d}(a_3) < \mathrm{d}(a_2) \\
a_2 & = & m_2\, a_3 & + & a_4 & 0 < \mathrm{d}(a_4) < \mathrm{d}(a_3) \\
\ldots & = & \ldots & + & & \\
a_{r-1} & = & m_{r-1}\, a_r & + & a_{r+1} & 0 < \mathrm{d}(a_{r+1}) < \mathrm{d}(a_r) \\
a_r & = & m_r\, a_{r+1} & + & 0 &
\end{array}
$$

The last non zero remainder is the gcd y. Here $y = a_{r+1}$.

The Euclid Algorithm is then extended to the case when $\mathrm{d}(a_1) < \mathrm{d}(a_2)$, by letting $\texttt{Euclid}(a_1, a_2) := \texttt{Euclid}(a_2, a_1)$

The combinatorial bijection induced by the Euclid Algorithm [$\ell = 2$]

$\texttt{Euclid}(a_1, a_2)$, [case $\mathrm{d}(a_1) \geq \mathrm{d}(a_2)$].

$$
\begin{array}{lcllll}
a_1 & = & m_1\, a_2 & + & a_3 & 0 < \mathrm{d}(a_3) < \mathrm{d}(a_2) \\
a_2 & = & m_2\, a_3 & + & a_4 & 0 < \mathrm{d}(a_4) < \mathrm{d}(a_3) \\
\dots & = & \dots & + & & \\
a_{r-1} & = & m_{r-1}\, a_r & + & a_{r+1} & 0 < \mathrm{d}(a_{r+1}) < \mathrm{d}(a_r) \\
a_r & = & m_r\, a_{r+1} & + & 0 &
\end{array}
$$

The last non zero remainder is the gcd y. Here $y = a_{r+1}$.

The Euclid Algorithm is then extended to the case when $\mathrm{d}(a_1) < \mathrm{d}(a_2)$, by letting $\texttt{Euclid}(a_1, a_2) := \texttt{Euclid}(a_2, a_1)$

The pair (a_1, a_2) of monic polynomials is entirely determined by

– the sequence of quotients $(m_1, m_2, \dots, m_r)$,

The combinatorial bijection induced by the Euclid Algorithm [$\ell = 2$]

$\texttt{Euclid}(a_1, a_2)$, [case $\mathrm{d}(a_1) \geq \mathrm{d}(a_2)$].

$$
\begin{array}{lcllll}
a_1 & = & m_1\, a_2 & + & a_3 & 0 < \mathrm{d}(a_3) < \mathrm{d}(a_2) \\
a_2 & = & m_2\, a_3 & + & a_4 & 0 < \mathrm{d}(a_4) < \mathrm{d}(a_3) \\
\ldots & = & \ldots & + & & \\
a_{r-1} & = & m_{r-1}\, a_r & + & a_{r+1} & 0 < \mathrm{d}(a_{r+1}) < \mathrm{d}(a_r) \\
a_r & = & m_r\, a_{r+1} & + & 0 &
\end{array}
$$

The last non zero remainder is the gcd y. Here $y = a_{r+1}$.

The Euclid Algorithm is then extended to the case when $\mathrm{d}(a_1) < \mathrm{d}(a_2)$, by letting $\texttt{Euclid}(a_1, a_2) := \texttt{Euclid}(a_2, a_1)$

The pair (a_1, a_2) of monic polynomials is entirely determined by

- the sequence of quotients $(m_1, m_2, \ldots, m_r)$, where
  - the first quotient m_1 is monic, with

$$\mathrm{d}(m_1) \geq 0 \ [\mathrm{d}(a_1) \geq \mathrm{d}(a_2)] \text{ or } \mathrm{d}(m_1) > 0 \ [\mathrm{d}(a_2) > \mathrm{d}(a_2)]$$

The combinatorial bijection induced by the Euclid Algorithm [$\ell = 2$]

$\texttt{Euclid}(a_1, a_2)$, [case $\mathrm{d}(a_1) \geq \mathrm{d}(a_2)$].

$$
\begin{array}{lcllll}
a_1 & = & m_1\, a_2 & + & a_3 & 0 < \mathrm{d}(a_3) < \mathrm{d}(a_2) \\
a_2 & = & m_2\, a_3 & + & a_4 & 0 < \mathrm{d}(a_4) < \mathrm{d}(a_3) \\
\ldots & = & \ldots & + & & \\
a_{r-1} & = & m_{r-1}\, a_r & + & a_{r+1} & 0 < \mathrm{d}(a_{r+1}) < \mathrm{d}(a_r) \\
a_r & = & m_r\, a_{r+1} & + & 0 &
\end{array}
$$

The last non zero remainder is the gcd y. Here $y = a_{r+1}$.

The Euclid Algorithm is then extended to the case when $\mathrm{d}(a_1) < \mathrm{d}(a_2)$, by letting $\texttt{Euclid}(a_1, a_2) := \texttt{Euclid}(a_2, a_1)$

The pair (a_1, a_2) of monic polynomials is entirely determined by

- the sequence of quotients $(m_1, m_2, \ldots, m_r)$, where
  - the first quotient m_1 is monic, with
    $\mathrm{d}(m_1) \geq 0$ $[\mathrm{d}(a_1) \geq \mathrm{d}(a_2)]$ or $\mathrm{d}(m_1) > 0$ $[\mathrm{d}(a_2) > \mathrm{d}(a_2)]$
  - any quotient m_i for $i \in [2..r]$ is general with $\mathrm{d}(m_i) > 0$

The combinatorial bijection induced by the Euclid Algorithm [$\ell = 2$]

$\texttt{Euclid}(a_1, a_2)$, [case $\mathrm{d}(a_1) \geq \mathrm{d}(a_2)$].

$$
\begin{array}{lcllll}
a_1 & = & m_1\, a_2 & + & a_3 & 0 < \mathrm{d}(a_3) < \mathrm{d}(a_2) \\
a_2 & = & m_2\, a_3 & + & a_4 & 0 < \mathrm{d}(a_4) < \mathrm{d}(a_3) \\
\ldots & = & \ldots & + & & \\
a_{r-1} & = & m_{r-1}\, a_r & + & a_{r+1} & 0 < \mathrm{d}(a_{r+1}) < \mathrm{d}(a_r) \\
a_r & = & m_r\, a_{r+1} & + & 0 &
\end{array}
$$

The last non zero remainder is the gcd y. Here $y = a_{r+1}$.

The Euclid Algorithm is then extended to the case when $\mathrm{d}(a_1) < \mathrm{d}(a_2)$, by letting $\texttt{Euclid}(a_1, a_2) := \texttt{Euclid}(a_2, a_1)$

The pair (a_1, a_2) of monic polynomials is entirely determined by

- the sequence of quotients $(m_1, m_2, \ldots, m_r)$, where
  - the first quotient m_1 is monic, with
    $\mathrm{d}(m_1) \geq 0$ [$\mathrm{d}(a_1) \geq \mathrm{d}(a_2)$] or $\mathrm{d}(m_1) > 0$ [$\mathrm{d}(a_2) > \mathrm{d}(a_2)$]
  - any quotient m_i for $i \in [2..r]$ is general with $\mathrm{d}(m_i) > 0$
- the monic gcd $y = a_{r+1}$.

Generating functions relative to the Euclid algorithm ($\ell = 2$).

Generating functions relative to the Euclid algorithm ($\ell = 2$).

$U(z)$ is the gen. function of the set $\mathcal{U}$ of monic polynomials

$G(z)$ is the gen. function of the general non constant polynomials

Generating functions relative to the Euclid algorithm ($\ell = 2$).

$U(z)$ is the gen. function of the set $\mathcal{U}$ of monic polynomials

$G(z)$ is the gen. function of the general non constant polynomials

$$U(z) = \sum_{a \in \mathcal{U}} z^{\mathrm{d}(a)} = \frac{1}{1 - qz}, \qquad G(z) = (q-1)\,[U(z) - 1] = \frac{(q-1)qz}{1 - qz}$$

Generating functions relative to the Euclid algorithm ($\ell = 2$).

$U(z)$ is the gen. function of the set $\mathcal{U}$ of monic polynomials

$G(z)$ is the gen. function of the general non constant polynomials

$$U(z) = \sum_{a \in \mathcal{U}} z^{\mathrm{d}(a)} = \frac{1}{1-qz}, \qquad G(z) = (q-1)\,[U(z)-1] = \frac{(q-1)qz}{1-qz}$$

$\Omega = \mathcal{U}^2$	$\approx$	{first quotient}	$\times$	{sequence of quotients}	$\times$	{GCD}
(a_1, a_2)	$\approx$	m_1	$\times$	$(m_2, \ldots, m_r)$	$\times$	y

Generating functions relative to the Euclid algorithm ($\ell = 2$).

$U(z)$ is the gen. function of the set $\mathcal{U}$ of monic polynomials

$G(z)$ is the gen. function of the general non constant polynomials

$$U(z) = \sum_{a \in \mathcal{U}} z^{\mathrm{d}(a)} = \frac{1}{1 - qz}, \qquad G(z) = (q-1)\,[U(z) - 1] = \frac{(q-1)qz}{1 - qz}$$

$$\begin{array}{rcccccc} \Omega = \mathcal{U}^2 & \approx & \{\text{first quotient}\} & \times & \{\text{sequence of quotients}\} & \times & \{\text{GCD}\} \\ (a_1, a_2) & \approx & m_1 & \times & (m_2, \ldots, m_r) & \times & y \\ U(z_1)\,U(z_2) & = & U(z_1) + [U(z_2) - 1] & \cdot & \dfrac{1}{1 - G(z_1 z_2)} & \cdot & U(z_1 z_2) \end{array}$$

Proof of the decomposition:

$$U(z_1)\,U(z_2) = U(z_1) + [U(z_2) - 1] \cdot \frac{1}{1 - G(z_1 z_2)} \cdot U(z_1 z_2)$$

Proof of the decomposition:

$$U(z_1)\,U(z_2) = U(z_1) + [U(z_2) - 1] \cdot \frac{1}{1 - G(z_1 z_2)} \cdot U(z_1 z_2)$$

There are two cases : (I) $\mathrm{d}(a_1) \geq \mathrm{d}(a_2)$ or (II) $\mathrm{d}(a_2) > \mathrm{d}(a_1)$

Proof of the decomposition:

$$U(z_1)\,U(z_2) = U(z_1) + [U(z_2) - 1] \cdot \frac{1}{1 - G(z_1 z_2)} \cdot U(z_1 z_2)$$

There are two cases : (I) $\mathrm{d}(a_1) \geq \mathrm{d}(a_2)$ or (II) $\mathrm{d}(a_2) > \mathrm{d}(a_1)$

`Euclid`(a_1, a_2), [case $\mathrm{d}(a_1) \geq \mathrm{d}(a_2)$].

$$\begin{array}{lcccc}
a_1 & = & m_1\, a_2 & + & a_3 \\
a_2 & = & m_2\, a_3 & + & a_4 \\
\dots & = & \dots & + & \dots \\
a_{r-1} & = & m_{r-1}\, a_r & + & a_{r+1} \\
a_r & = & m_r\, y & + & 0
\end{array}
\qquad
\begin{array}{c}
\mathrm{d}(a_1) = \mathrm{d}(m_1) + \mathrm{d}(a_2) \\
\mathrm{d}(a_2) = \mathrm{d}(m_2) + \mathrm{d}(a_3) \\
\dots \\
\mathrm{d}(a_{r-1}) = \mathrm{d}(m_{r-1}) + \mathrm{d}(a_r) \\
\mathrm{d}(a_r) = \mathrm{d}(m_r) + \mathrm{d}(y)
\end{array}$$

Proof of the decomposition:

$$U(z_1)\,U(z_2) = U(z_1) + [U(z_2) - 1] \cdot \frac{1}{1 - G(z_1 z_2)} \cdot U(z_1 z_2)$$

There are two cases : (I) $\mathrm{d}(a_1) \geq \mathrm{d}(a_2)$ or (II) $\mathrm{d}(a_2) > \mathrm{d}(a_1)$

$\texttt{Euclid}(a_1, a_2)$, [case $\mathrm{d}(a_1) \geq \mathrm{d}(a_2)$].

$$\begin{array}{lcllll}
a_1 & = & m_1\, a_2 & + & a_3 & \mathrm{d}(a_1) = \mathrm{d}(m_1) + \mathrm{d}(a_2) \\
a_2 & = & m_2\, a_3 & + & a_4 & \mathrm{d}(a_2) = \mathrm{d}(m_2) + \mathrm{d}(a_3) \\
\ldots & = & \ldots & + & \ldots & \ldots \\
a_{r-1} & = & m_{r-1}\, a_r & + & a_{r+1} & \mathrm{d}(a_{r-1}) = \mathrm{d}(m_{r-1}) + \mathrm{d}(a_r) \\
a_r & = & m_r\, y & + & 0 & \mathrm{d}(a_r) = \mathrm{d}(m_r) + \mathrm{d}(y)
\end{array}$$

(I)
$$\left|\begin{array}{rcllll}
\mathrm{d}(a_1) & = & \mathrm{d}(m_1) & + & \mathrm{d}(m_2) + \ldots + \mathrm{d}(m_r) & + \quad \mathrm{d}(y) \\
\mathrm{d}(a_2) & = & & & \mathrm{d}(m_2) + \ldots + \mathrm{d}(m_r) & + \quad \mathrm{d}(y) \\
z_1^{\mathrm{d}(a_1)}\, z_2^{\mathrm{d}(a_2)} & = & z_1^{\mathrm{d}(m_1)} & \cdot & (z_1 z_2)^{\mathrm{d}(m_2)+\ldots+\mathrm{d}(m_r)} & \cdot \quad (z_1 z_2)^{\mathrm{d}(y)}
\end{array}\right.$$

Proof of the decomposition:

$$U(z_1)\,U(z_2) = U(z_1) + [U(z_2) - 1] \cdot \frac{1}{1 - G(z_1 z_2)} \cdot U(z_1 z_2)$$

There are two cases : (I) $\mathrm{d}(a_1) \geq \mathrm{d}(a_2)$ or (II) $\mathrm{d}(a_2) > \mathrm{d}(a_1)$

Euclid(a_1, a_2), [case $\mathrm{d}(a_1) \geq \mathrm{d}(a_2)$].

$$\begin{array}{lcccll}
a_1 & = & m_1\, a_2 & + & a_3 & \mathrm{d}(a_1) = \mathrm{d}(m_1) + \mathrm{d}(a_2) \\
a_2 & = & m_2\, a_3 & + & a_4 & \mathrm{d}(a_2) = \mathrm{d}(m_2) + \mathrm{d}(a_3) \\
\ldots & = & \ldots & + & \ldots & \ldots \\
a_{r-1} & = & m_{r-1}\, a_r & + & a_{r+1} & \mathrm{d}(a_{r-1}) = \mathrm{d}(m_{r-1}) + \mathrm{d}(a_r) \\
a_r & = & m_r\, y & + & 0 & \mathrm{d}(a_r) = \mathrm{d}(m_r) + \mathrm{d}(y)
\end{array}$$

(I)

$$\begin{array}{ccccccc}
\mathrm{d}(a_1) & = & \mathrm{d}(m_1) & + & \mathrm{d}(m_2) + \ldots + \mathrm{d}(m_r) & + & \mathrm{d}(y) \\
\mathrm{d}(a_2) & = & & & \mathrm{d}(m_2) + \ldots + \mathrm{d}(m_r) & + & \mathrm{d}(y) \\
z_1^{\mathrm{d}(a_1)}\, z_2^{\mathrm{d}(a_2)} & = & z_1^{\mathrm{d}(m_1)} & \cdot & (z_1 z_2)^{\mathrm{d}(m_2)+\ldots+\mathrm{d}(m_r)} & \cdot & (z_1 z_2)^{\mathrm{d}(y)}
\end{array}$$

(II)

$$\begin{array}{ccccccc}
\mathrm{d}(a_1) & = & & & \mathrm{d}(m_2) + \ldots + \mathrm{d}(m_r) & + & \mathrm{d}(y) \\
\mathrm{d}(a_2) & = & \mathrm{d}(m_1) & + & \mathrm{d}(m_2) + \ldots + \mathrm{d}(m_r) & + & \mathrm{d}(y) \\
z_1^{\mathrm{d}(a_1)}\, z_2^{\mathrm{d}(a_2)} & = & z_2^{\mathrm{d}(m_1)} & \cdot & (z_1 z_2)^{\mathrm{d}(m_2)+\ldots+\mathrm{d}(m_r)} & \cdot & (z_1 z_2)^{\mathrm{d}(y)}
\end{array}$$

Algorithmic expression of the GF of the ℓ-Euclid Algorithm

We have shown that the Euclid algorithm ($\ell = 2$) translates as a product

$$U(z_1)\,U(z_2) = T(z_1, z_2)\;U(z_1 z_2), \qquad \text{with} \quad T(z_1, z_2) = \frac{U(z_1) + U(z_2) - 1}{1 - G(z_1 z_2)}$$

Algorithmic expression of the GF of the ℓ-Euclid Algorithm

We have shown that the Euclid algorithm ($\ell = 2$) translates as a product

$$U(z_1)\,U(z_2) = T(z_1, z_2)\; U(z_1 z_2), \qquad \text{with} \quad T(z_1, z_2) = \frac{U(z_1) + U(z_2) - 1}{1 - G(z_1 z_2)}$$

Then, for any $\ell \geq 2$, the ℓ–Euclid algorithm translates as the product

$$U(z_1) \cdot \ldots \cdot U(z_\ell) = U(t_\ell) \prod_{k=1}^{\ell-1} T(z_{k+1}, t_k) \qquad [t_k := z_1 \cdot z_2 \cdot \ldots \cdot z_k]$$

Algorithmic expression of the GF of the ℓ-Euclid Algorithm

We have shown that the Euclid algorithm ($\ell = 2$) translates as a product

$$U(z_1)\,U(z_2) = T(z_1, z_2)\; U(z_1 z_2), \qquad \text{with} \quad T(z_1, z_2) = \frac{U(z_1) + U(z_2) - 1}{1 - G(z_1 z_2)}$$

Then, for any $\ell \geq 2$, the ℓ–Euclid algorithm translates as the product

$$U(z_1) \cdot \ldots \cdot U(z_\ell) = U(t_\ell) \prod_{k=1}^{\ell-1} T(z_{k+1}, t_k) \qquad [t_k := z_1 \cdot z_2 \cdot \ldots \cdot z_k]$$

Now, with $z = z_1 = \ldots = z_\ell$,
the (plain) generating function $S(z)$ of $\mathcal{U}^\ell$ has the alternative expression

$$S(z) = U(z)^\ell = U(z^\ell) \prod_{k=1}^{\ell-1} T(z, z^k)$$

which is an exact translation of the ℓ-Euclid algorithm.

T is the "phase generating function".

Bivariate Generating Functions relative to the ℓ-Euclid Algorithm

We start with: $$S(z) = U(z)^{\ell} = U(z^{\ell}) \prod_{k=1}^{\ell-1} T(z, z^k)$$

Bivariate Generating Functions relative to the ℓ-Euclid Algorithm

We start with: $$S(z) = U(z)^\ell = U(z^\ell) \prod_{k=1}^{\ell-1} T(z, z^k)$$

For studying the distribution of the two costs :

- L_k (number of steps in the k-th phase)
- D_k (degree of the gcd at the beginning of the k-th phase)

we use bivariate generating functions,

with an extra variable u which marks the cost

Bivariate Generating Functions relative to the ℓ-Euclid Algorithm

We start with: $$S(z) = U(z)^\ell = U(z^\ell) \prod_{k=1}^{\ell-1} T(z, z^k)$$

For studying the distribution of the two costs :

– L_k (number of steps in the k-th phase)

– D_k (degree of the gcd at the beginning of the k-th phase)

we use bivariate generating functions,

with an extra variable u which marks the cost

$$L_k(z,u) = U(z)^\ell \cdot \frac{T(z, z^k, u)}{T(z, z^k)}, \qquad D_k(z,u) = U(z)^\ell \cdot \frac{U(z^k, u)}{U(z^k)},$$

With:

$$T(z,t,u) = u\frac{U(z) + U(t) - 1}{1 - uG(zt)}, \qquad U(t,u) = \frac{1}{1 - qut}.$$

Towards the distributional analysis of L_k and D_k.

$$\mathbb{P}_n[L_k > m] = \sum_{j>m} \frac{[u^j z^n] L_k(z,u)}{[z^n]S(z)} = \frac{[z^n]\sum_{j>m}[u^j]L_k(z,u)}{[z^n]S(z)} = \frac{[z^n]\widehat{L}_k^{[m]}(z)}{[z^n]S(z)}$$

$$\mathbb{P}_n[D_k > m] = \sum_{j>m} \frac{[u^j z^n] D_k(z,u)}{[z^n]S(z)} = \frac{[z^n]\sum_{j>m}[u^j]D_k(z,u)}{[z^n]S(z)} = \frac{[z^n]\widehat{D}_k^{[m]}(z)}{[z^n]S(z)}$$

Towards the distributional analysis of L_k and D_k.

$$\mathbb{P}_n[L_k > m] = \sum_{j>m} \frac{[u^j z^n] L_k(z,u)}{[z^n]S(z)} = \frac{[z^n]\sum_{j>m}[u^j]L_k(z,u)}{[z^n]S(z)} = \frac{[z^n]\widehat{L}_k^{[m]}(z)}{[z^n]S(z)}$$

$$\mathbb{P}_n[D_k > m] = \sum_{j>m} \frac{[u^j z^n] D_k(z,u)}{[z^n]S(z)} = \frac{[z^n]\sum_{j>m}[u^j]D_k(z,u)}{[z^n]S(z)} = \frac{[z^n]\widehat{D}_k^{[m]}(z)}{[z^n]S(z)}$$

The generating functions "of the numerators" are

$$\widehat{L}_k^{[m]}(z) = \frac{1}{(1-qz)^\ell} \cdot G(z^{k+1})^m, \qquad \widehat{D}_k^{[m]}(z) = \frac{1}{(1-qz)^\ell} \cdot (qz^k)^m,$$

both of type $\dfrac{1}{(1-qz)^\ell} \cdot A_k(z)^m,$

Towards the distributional analysis of L_k and D_k.

$$\mathbb{P}_n[L_k > m] = \sum_{j>m} \frac{[u^j z^n] L_k(z,u)}{[z^n]S(z)} = \frac{[z^n]\sum_{j>m}[u^j]L_k(z,u)}{[z^n]S(z)} = \frac{[z^n]\widehat{L}_k^{[m]}(z)}{[z^n]S(z)}$$

$$\mathbb{P}_n[D_k > m] = \sum_{j>m} \frac{[u^j z^n] D_k(z,u)}{[z^n]S(z)} = \frac{[z^n]\sum_{j>m}[u^j]D_k(z,u)}{[z^n]S(z)} = \frac{[z^n]\widehat{D}_k^{[m]}(z)}{[z^n]S(z)}$$

The generating functions "of the numerators" are

$$\widehat{L}_k^{[m]}(z) = \frac{1}{(1-qz)^\ell} \cdot G(z^{k+1})^m, \qquad \widehat{D}_k^{[m]}(z) = \frac{1}{(1-qz)^\ell} \cdot (qz^k)^m,$$

$$\text{both of type} \quad \frac{1}{(1-qz)^\ell} \cdot A_k(z)^m,$$

The asymptotics depends on the value $a := A_k(1/q)$ at the pole $z = 1/q$

- For the first phase $k = 1$, one has $a = 1$
- For the subsequent phases $k \geq 2$, one has $a < 1$

A general result for the asymptotics of coefficients

Consider the function $F^{[m]}(z) = \dfrac{1}{(1-z)^{\ell}}\, A(z)^m, \quad$ with $\ell \geq 2$

where (i) $A(z)$ is analytic on the disk $|z| \leq \rho$ with $\rho > 1$,

(ii) $a := A(1) \neq 0, \quad b := A'(1) > 0$,

(iii) for $|z|$ close enough to 1, $|A(z)| \leq A(|z|)$.

Then, for $n \to \infty$ and $m/n \in [0, a/b[$,

$$[z^n]F^{[m]}(z) = \frac{n^{\ell-1}}{(\ell-1)!}\, a^m \left(1 - \frac{b}{a}\frac{m}{n}\right)^{\ell-1} \left[1 + O\left(\frac{1}{n}\right)\right].$$

A general result for the asymptotics of coefficients

Consider the function $F^{[m]}(z) = \dfrac{1}{(1-z)^\ell}\, A(z)^m, \quad$ with $\ell \geq 2$

where (i) $A(z)$ is analytic on the disk $|z| \leq \rho$ with $\rho > 1$,

(ii) $a := A(1) \neq 0, \quad b := A'(1) > 0,$

(iii) for $|z|$ close enough to 1, $|A(z)| \leq A(|z|)$.

Then, for $n \to \infty$ and $m/n \in [0, a/b[$,

$$[z^n]F^{[m]}(z) = \frac{n^{\ell-1}}{(\ell-1)!}\, a^m \left(1 - \frac{b}{a}\frac{m}{n}\right)^{\ell-1} \left[1 + O\left(\frac{1}{n}\right)\right].$$

Application to the present situation.

For the first phase: $a = 1 \implies$ A "beta" behavior $(1, \ell - 1)$

For the subsequent phases: $a < 1 \implies$ A geometric behavior of ratio a

Main result for the number of divisions L_k – First phase $(k = 1)$

The number of divisions L_1 performed during the first phase

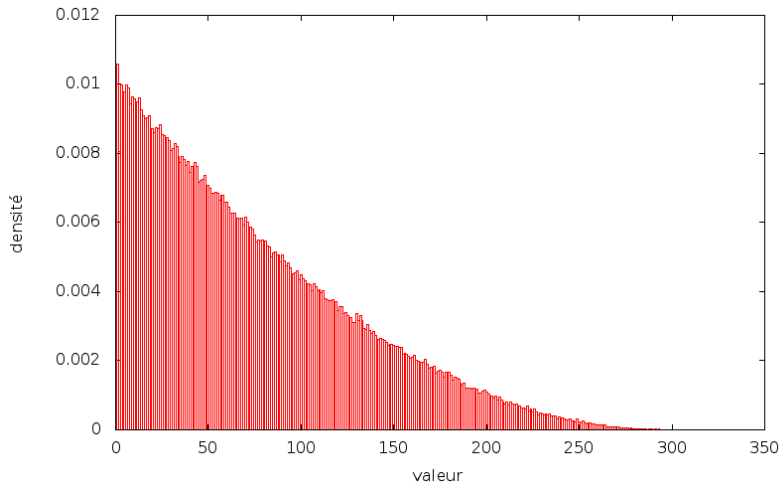
– has a mean value of linear order

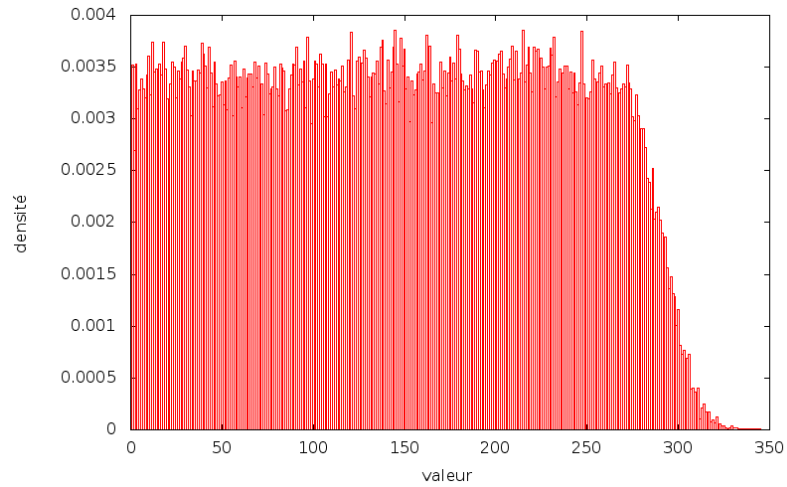
$$\mathbb{E}_n[L_1] = \frac{q-1}{2q}\frac{n}{\ell} + \frac{3q+1}{4q} + O\left(\frac{1}{n}\right). \qquad \frac{2q}{q-1} = \text{entropy}$$

– follows an asymptotic beta law of parameter $(1, \ell - 1)$.

Its distribution satisfies when $n \to \infty$, and $m/n \in [0, (q-1)/(2q)]$

$$\mathbb{P}_n[L_1 > m] = \left(1 - \frac{2q}{q-1}\frac{m}{n}\right)^{\ell-1} + O\left(\frac{1}{n^\alpha}\right).$$





Main result for the number of divisions L_k – Subsequent phases (case $k \geq 2$)

For $k \geq 2$, the number of divisions performed during the k-th phase

– has a mean value of constant order

$$\mathbb{E}_n[L_k] = \left(1 + \frac{q-1}{q^k - q}\right) + O\left(\frac{1}{n}\right),$$

– follows an asymptotic geometric law, with ratio $\dfrac{q-1}{q^k-1}$

For $n \to \infty$ and $m/n \in [0, 1/(k+1) \cdot (q^k-1)/q^k]$

$$\mathbb{P}_n[L_k \geq m] = \left(\frac{q-1}{q^k-1}\right)^m + O\left(\frac{\log n}{n}\right)$$

Main result for the number of divisions L_k – Subsequent phases (case $k \geq 2$)

For $k \geq 2$, the number of divisions performed during the k-th phase

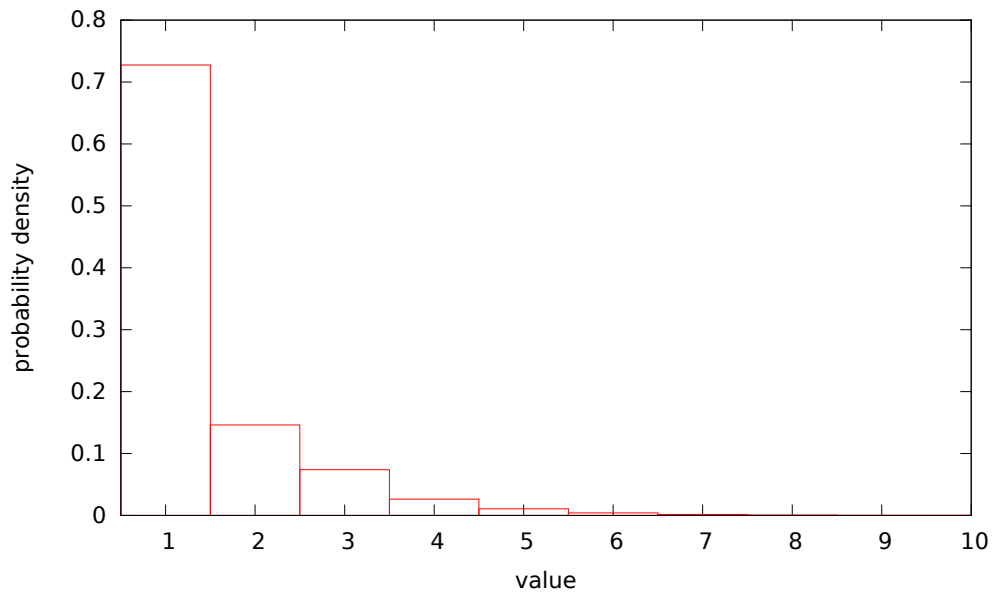
– has a mean value of constant order

$$\mathbb{E}_n[L_k] = \left(1 + \frac{q-1}{q^k - q}\right) + O\left(\frac{1}{n}\right),$$

– follows an asymptotic geometric law, with ratio $\dfrac{q-1}{q^k-1}$

For $n \to \infty$ and $m/n \in [0, 1/(k+1) \cdot (q^k-1)/q^k]$

$$\mathbb{P}_n[L_k \geq m] = \left(\frac{q-1}{q^k-1}\right)^m + O\left(\frac{\log n}{n}\right)$$



III. Analysis on integers.

III. Analysis on integers.

We use Dirichlet generating functions
together with a dynamical point of view

Remind : The Euclid algorithm ($\ell = 2$) on polynomials

translates as a product of power generating functions

$$U(z)\,U(t) = U(zt) \cdot \frac{1}{1 - G(zt)} \cdot [U(z) + U(t) - 1].$$

Remind : The Euclid algorithm ($\ell = 2$) on polynomials

translates as a product of power generating functions

$$U(z)\,U(t) = U(zt) \cdot \frac{1}{1 - G(zt)} \cdot [U(z) + U(t) - 1].$$

We will prove that the Euclid algorithm ($\ell = 2$) on integers

translates as a product of Dirichlet generating functions

$$\zeta(s)\,\zeta(t) = \zeta(s+t) \cdot \left(\frac{1}{2} (I - \mathbf{G}_{s+t})^{-1} \circ (\mathbf{G}_s + \mathbf{G}_t) \right) [1](0)$$

Remind : The Euclid algorithm ($\ell = 2$) on polynomials

translates as a product of power generating functions

$$U(z)\,U(t) = U(zt) \cdot \frac{1}{1 - G(zt)} \cdot [U(z) + U(t) - 1].$$

We will prove that the Euclid algorithm ($\ell = 2$) on integers

translates as a product of Dirichlet generating functions

$$\zeta(s)\,\zeta(t) = \zeta(s+t) \cdot \left(\frac{1}{2} (I - \mathbf{G}_{s+t})^{-1} \circ (\mathbf{G}_s + \mathbf{G}_t) \right) [1](0)$$

This involves – the Riemann Dirichlet series $\zeta(s) = \sum_{a \geq 1} a^{-s}$

– a functional operator $\mathbf{G}_s$ (that depends on a parameter s)

$$\mathbf{G}_s[f](t) = \sum_{m \geq 1} \left(\frac{1}{m+t} \right)^s f\left(\frac{1}{m+t} \right)$$

Remind : The Euclid algorithm ($\ell = 2$) on polynomials

translates as a product of power generating functions

$$U(z)\,U(t) = U(zt) \cdot \frac{1}{1 - G(zt)} \cdot [U(z) + U(t) - 1].$$

We will prove that the Euclid algorithm ($\ell = 2$) on integers

translates as a product of Dirichlet generating functions

$$\zeta(s)\,\zeta(t) = \zeta(s+t) \cdot \left(\frac{1}{2} (I - \mathbf{G}_{s+t})^{-1} \circ (\mathbf{G}_s + \mathbf{G}_t) \right) [1](0)$$

This involves – the Riemann Dirichlet series $\zeta(s) = \sum_{a \geq 1} a^{-s}$

– a functional operator $\mathbf{G}_s$ (that depends on a parameter s)

$$\mathbf{G}_s[f](t) = \sum_{m \geq 1} \left(\frac{1}{m+t} \right)^s f\left(\frac{1}{m+t} \right)$$

The operator $\mathbf{G}_s$ "generates" the quotients; it is closely related to the transfer operator of the underlying dynamical system...

Remind : The Euclid algorithm ($\ell = 2$) on polynomials

translates as a product of power generating functions

$$U(z)\,U(t) = U(zt) \cdot \frac{1}{1 - G(zt)} \cdot [U(z) + U(t) - 1].$$

We will prove that the Euclid algorithm ($\ell = 2$) on integers

translates as a product of Dirichlet generating functions

$$\zeta(s)\,\zeta(t) = \zeta(s+t) \cdot \left(\frac{1}{2} (I - \mathbf{G}_{s+t})^{-1} \circ (\mathbf{G}_s + \mathbf{G}_t) \right) [1](0)$$

This involves – the Riemann Dirichlet series $\zeta(s) = \sum_{a \geq 1} a^{-s}$

– a functional operator $\mathbf{G}_s$ (that depends on a parameter s)

$$\mathbf{G}_s[f](t) = \sum_{m \geq 1} \left(\frac{1}{m+t} \right)^s f\left(\frac{1}{m+t} \right)$$

The operator $\mathbf{G}_s$ "generates" the quotients; it is closely related to the transfer operator of the underlying dynamical system...

An instance of a "dynamical" analysis....

More involved than the previous one, but provides the same type of results.

Similarities and differences between the two analyses

	Polynomials	Integers
GF	Power GF's	Dirichlet GF and operators
Basic tool	$G(z) = \sum_m z^{\mathrm{d}(m)}$	$\mathbf{G}_s[f](t) = \sum_{m \geq 1} \left(\frac{1}{m+t}\right)^s f\left(\frac{1}{m+t}\right)$
Phase GF	$\frac{U(z^k) + U(z) - 1}{1 - G(z^{k+1})}$	$(I - \mathbf{G}_{(k+1)s})^{-1} \circ (\mathbf{G}_{ks} + \mathbf{G}_s)[1](0)$
Singularities	z s.t. $G(z^{k+1}) = 1$	s s.t. $\lambda((k+1)s) = 1$
Extraction	Cauchy Formula	Perron Formula
Contours	Disks	Vertical lines

Similarities and differences between the two analyses

	Polynomials	Integers
GF	Power GF's	Dirichlet GF and operators
Basic tool	$G(z) = \sum_m z^{\mathrm{d}(m)}$	$\mathbf{G}_s[f](t) = \sum_{m\geq 1} \left(\frac{1}{m+t}\right)^s f\left(\frac{1}{m+t}\right)$
Phase GF	$\frac{U(z^k)+U(z)-1}{1-G(z^{k+1})}$	$(I-\mathbf{G}_{(k+1)s})^{-1} \circ (\mathbf{G}_{ks}+\mathbf{G}_s)[1](0)$
Singularities	z s.t. $G(z^{k+1})=1$	s s.t. $\lambda((k+1)s)=1$
Extraction Contours	Cauchy Formula Disks	Perron Formula Vertical lines

$\lambda(s)$ is the dominant eigenvalue of $\mathbf{G}_s$

$\lambda(2)=1$; $\lambda'(2)$ closely related to the entropy

Results in the integer case.

NB: the integer size of the input $\approx$ number of digits in base e

We prove the following facts about the number of divisions performed

Results in the integer case.

NB: the integer size of the input $\approx$ number of digits in base e

We prove the following facts about the number of divisions performed

- during the first phase:
  - it is linear on average,
  - it asymptotically follows a beta law;

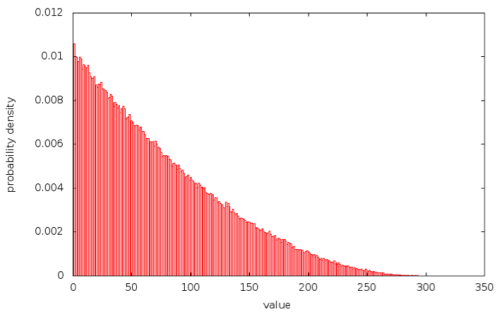
Results in the integer case.

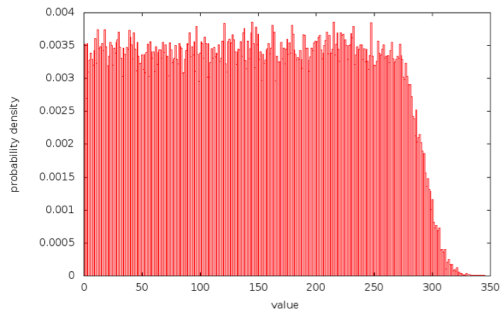
NB: the integer size of the input $\approx$ number of digits in base e

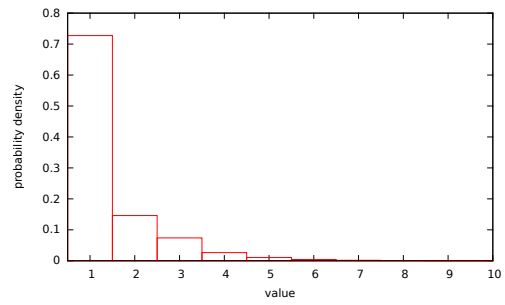
We prove the following facts about the number of divisions performed

- during the first phase:
  - it is linear on average,
  - it asymptotically follows a beta law;
- during subsequent phase:
  - it is constant on average
  - it asymptotically follows a geometric law

The same phenomena occur for the size of the partial gcd.







Probabilistic analysis of the plain ℓ–GCD algorithm on $\mathbb{Z}$.

On the input $(x_1, x_2, \ldots, x_\ell)$,

- the algorithm computes the total gcd $y_\ell := \gcd(x_1, x_2, \ldots, x_\ell)$
- with $\ell - 1$ phases.
- The k-th phase computes the k–th gcd,

$$y_k := \gcd(x_k, y_{k-1}) = \gcd(x_1, x_2, \ldots, x_k) \ .$$

Probabilistic analysis of the plain ℓ–GCD algorithm on $\mathbb{Z}$.

On the input $(x_1, x_2, \ldots, x_\ell)$,

- the algorithm computes the total gcd $y_\ell := \gcd(x_1, x_2, \ldots, x_\ell)$
- with $\ell - 1$ phases.
- The k-th phase computes the k–th gcd,

$$y_k := \gcd(x_k, y_{k-1}) = \gcd(x_1, x_2, \ldots, x_k) \ .$$

- each phase performs the classical Euclid algorithm

via a sequence of Euclidean divisions

Probabilistic analysis of the plain ℓ–GCD algorithm on $\mathbb{Z}$.

On the input $(x_1, x_2, \ldots, x_\ell)$,

- the algorithm computes the total gcd $y_\ell := \gcd(x_1, x_2, \ldots, x_\ell)$
- with $\ell - 1$ phases.
- The k-th phase computes the k–th gcd,

$$y_k := \gcd(x_k, y_{k-1}) = \gcd(x_1, x_2, \ldots, x_k) \ .$$

- each phase performs the classical Euclid algorithm

via a sequence of Euclidean divisions

The set of inputs is $\Omega = \{\underline{x} := (x_1, \ldots, x_\ell); \quad x_i \in \mathbb{N}\}$

The size of an input: $\mathrm{d}(\underline{x}) := \mathrm{d}(x_1 x_2 \ldots x_\ell)$ with $\mathrm{d}(x) := \lfloor \log x \rfloor$

"almost additive" $\mathrm{d}(\underline{x}) \approx \mathrm{d}(x_1) + \ldots + \mathrm{d}(x_\ell)$

Probabilistic analysis of the plain ℓ–GCD algorithm on $\mathbb{Z}$.

On the input $(x_1, x_2, \ldots, x_\ell)$,

- the algorithm computes the total gcd $y_\ell := \gcd(x_1, x_2, \ldots, x_\ell)$
- with $\ell - 1$ phases.
- The k-th phase computes the k–th gcd,

$$y_k := \gcd(x_k, y_{k-1}) = \gcd(x_1, x_2, \ldots, x_k) \ .$$

- each phase performs the classical Euclid algorithm

via a sequence of Euclidean divisions

The set of inputs is $\Omega = \{\underline{x} := (x_1, \ldots, x_\ell); \quad x_i \in \mathbb{N}\}$

The size of an input: $\mathrm{d}(\underline{x}) := \mathrm{d}(x_1 x_2 \ldots x_\ell)$ with $\mathrm{d}(x) := \lfloor \log x \rfloor$

"almost additive" $\mathrm{d}(\underline{x}) \approx \mathrm{d}(x_1) + \ldots + \mathrm{d}(x_\ell)$

Main costs of interest

- the number L_k of divisions during the k–th phase

i.e. on the input (x_k, y_{k-1})

- the size D_k of the k–th gcd

(at the beginning of the k-th phase).

The combinatorial bijection induced by the Euclid Algorithm $[\ell = 2]$

The combinatorial bijection induced by the Euclid Algorithm [$\ell = 2$]

`Euclid`(a_1, a_2), [case $a_1 \geq a_2$].

$$
\begin{array}{lcllll}
a_1 & = & m_1\, a_2 & + & a_3 & 0 < a_3 < a_2 \\
a_2 & = & m_2\, a_3 & + & a_4 & 0 < \ a_4 < a_3 \\
\ldots & = & \ldots & + & & \\
a_{r-1} & = & m_{r-1}\, a_r & + & a_{r+1} & 0 < a_{r+1} < a_r \\
a_r & = & m_r\, a_{r+1} & + & 0 &
\end{array}
$$

The last non zero remainder is the gcd y. Here $y = a_{r+1}$.

The combinatorial bijection induced by the Euclid Algorithm $[\ell = 2]$

$\texttt{Euclid}(a_1, a_2)$, [case $a_1 \geq a_2$].

$$
\begin{array}{lcllll}
a_1 & = & m_1\, a_2 & + & a_3 & 0 < a_3 < a_2 \\
a_2 & = & m_2\, a_3 & + & a_4 & 0 < \ a_4 < a_3 \\
\ldots & = & \ldots & + & & \\
a_{r-1} & = & m_{r-1}\, a_r & + & a_{r+1} & 0 < a_{r+1} < a_r \\
a_r & = & m_r\, a_{r+1} & + & 0 &
\end{array}
$$

The last non zero remainder is the gcd y. Here $y = a_{r+1}$.

The Euclid Algorithm is then extended to the case when $a_1 < a_2$, by letting $\texttt{Euclid}(a_1, a_2) := \texttt{Euclid}(a_2, a_1)$

The combinatorial bijection induced by the Euclid Algorithm [$\ell = 2$]

$\texttt{Euclid}(a_1, a_2)$, [case $a_1 \geq a_2$].

$$
\begin{array}{lcllll}
a_1 & = & m_1\, a_2 & + & a_3 & 0 < a_3 < a_2 \\
a_2 & = & m_2\, a_3 & + & a_4 & 0 < a_4 < a_3 \\
\ldots & = & \ldots & + & & \\
a_{r-1} & = & m_{r-1}\, a_r & + & a_{r+1} & 0 < a_{r+1} < a_r \\
a_r & = & m_r\, a_{r+1} & + & 0 &
\end{array}
$$

The last non zero remainder is the gcd y. Here $y = a_{r+1}$.

The Euclid Algorithm is then extended to the case when $a_1 < a_2$, by letting $\texttt{Euclid}(a_1, a_2) := \texttt{Euclid}(a_2, a_1)$

The pair (a_1, a_2) of positive integers is entirely determined by

– the sequence of quotients $(m_1, m_2, \ldots, m_r)$,

The combinatorial bijection induced by the Euclid Algorithm $[\ell = 2]$

$\texttt{Euclid}(a_1, a_2)$, [case $a_1 \geq a_2$].

$$
\begin{array}{lcllll}
a_1 & = & m_1\, a_2 & + & a_3 & 0 < a_3 < a_2 \\
a_2 & = & m_2\, a_3 & + & a_4 & 0 < \ a_4 < a_3 \\
\ldots & = & \ldots & + & & \\
a_{r-1} & = & m_{r-1}\, a_r & + & a_{r+1} & 0 < a_{r+1} < a_r \\
a_r & = & m_r\, a_{r+1} & + & 0 &
\end{array}
$$

The last non zero remainder is the gcd y. Here $y = a_{r+1}$.

The Euclid Algorithm is then extended to the case when $a_1 < a_2$, by letting $\texttt{Euclid}(a_1, a_2) := \texttt{Euclid}(a_2, a_1)$

The pair (a_1, a_2) of positive integers is entirely determined by

– the sequence of quotients $(m_1, m_2, \ldots, m_r)$, where

– the first quotient m_1 satisfies

$m_1 \geq 0$ $[a_1 \geq a_2]$ or $m_1 \geq 1$ $[a_1 < a_2]$

The combinatorial bijection induced by the Euclid Algorithm [$\ell = 2$]

$\texttt{Euclid}(a_1, a_2)$, [case $a_1 \geq a_2$].

$$
\begin{array}{lcllll}
a_1 & = & m_1\, a_2 & + & a_3 & 0 < a_3 < a_2 \\
a_2 & = & m_2\, a_3 & + & a_4 & 0 < \ a_4 < a_3 \\
\ldots & = & \ldots & + & & \\
a_{r-1} & = & m_{r-1}\, a_r & + & a_{r+1} & 0 < a_{r+1} < a_r \\
a_r & = & m_r\, a_{r+1} & + & 0 &
\end{array}
$$

The last non zero remainder is the gcd y. Here $y = a_{r+1}$.

The Euclid Algorithm is then extended to the case when $a_1 < a_2$, by letting $\texttt{Euclid}(a_1, a_2) := \texttt{Euclid}(a_2, a_1)$

The pair (a_1, a_2) of positive integers is entirely determined by

- the sequence of quotients $(m_1, m_2, \ldots, m_r)$, where
  - the first quotient m_1 satisfies
    $$m_1 \geq 0 \ [a_1 \geq a_2] \text{ or } m_1 \geq 1 \ [a_1 < a_2]$$
  - any quotient m_i for $i \in [2..r]$ satisfies $m_i \geq 1$

The combinatorial bijection induced by the Euclid Algorithm [$\ell = 2$]

$\texttt{Euclid}(a_1, a_2)$, [case $a_1 \geq a_2$].

$$
\begin{array}{lcllll}
a_1 & = & m_1\, a_2 & + & a_3 & 0 < a_3 < a_2 \\
a_2 & = & m_2\, a_3 & + & a_4 & 0 < a_4 < a_3 \\
\ldots & = & \ldots & + & & \\
a_{r-1} & = & m_{r-1}\, a_r & + & a_{r+1} & 0 < a_{r+1} < a_r \\
a_r & = & m_r\, a_{r+1} & + & 0 &
\end{array}
$$

The last non zero remainder is the gcd y. Here $y = a_{r+1}$.

The Euclid Algorithm is then extended to the case when $a_1 < a_2$, by letting $\texttt{Euclid}(a_1, a_2) := \texttt{Euclid}(a_2, a_1)$

The pair (a_1, a_2) of positive integers is entirely determined by

- the sequence of quotients $(m_1, m_2, \ldots, m_r)$, where
  - the first quotient m_1 satisfies
    $$m_1 \geq 0 \ [a_1 \geq a_2] \text{ or } m_1 \geq 1 \ [a_1 < a_2]$$
  - any quotient m_i for $i \in [2..r]$ satisfies $m_i \geq 1$
- the gcd $y = a_{r+1}$ satisfies $y \geq 1$

Euclid Algorithm and Continued Fractions Expansions

We use Continued Fractions Expansions and LFT's $h_m(x) := 1/(m+x)$

Euclid Algorithm and Continued Fractions Expansions

We use Continued Fractions Expansions and LFT's $h_m(x) := 1/(m+x)$

`Euclid`(a_1, a_2), [case $a_1 \geq a_2$].

Euclid Algorithm and Continued Fractions Expansions

We use Continued Fractions Expansions and LFT's $h_m(x) := 1/(m+x)$

`Euclid`(a_1, a_2), [case $a_1 \geq a_2$].

$$y := \gcd(a_1, a_2); \qquad a_1 = u_1\, y, \quad a_2 = u_2\, y, \quad a_3 = u_3\, y$$

$$\gcd(u_1, u_2) = 1, \gcd(u_2, u_3) = 1$$

Euclid Algorithm and Continued Fractions Expansions

We use Continued Fractions Expansions and LFT's $h_m(x) := 1/(m+x)$

`Euclid`(a_1, a_2), [case $a_1 \geq a_2$].

$$y := \gcd(a_1, a_2); \qquad a_1 = u_1\, y, \quad a_2 = u_2\, y, \quad a_3 = u_3\, y$$

$$\gcd(u_1, u_2) = 1, \gcd(u_2, u_3) = 1$$

$$\frac{u_2}{u_1} = \cfrac{1}{m_1 + \cfrac{1}{m_2 + \cfrac{1}{m_3 + \cfrac{1}{\ddots + \cfrac{1}{m_r}}}}}, \qquad \frac{u_3}{u_2} = \cfrac{1}{m_2 + \cfrac{1}{m_3 + \cfrac{1}{\ddots + \cfrac{1}{m_r}}}},$$

$$\frac{u_2}{u_1} = h_{m_1} \circ h_{m_2} \circ \ldots \circ h_{m_r}(0), \qquad \frac{u_3}{u_2} = h_{m_2} \circ \ldots \circ h_{m_r}(0)$$

Euclid Algorithm and Continued Fractions Expansions

We use Continued Fractions Expansions and LFT's $h_m(x) := 1/(m+x)$

$\texttt{Euclid}(a_1, a_2)$, [case $a_1 \geq a_2$].

$$y := \gcd(a_1, a_2); \qquad a_1 = u_1\, y, \quad a_2 = u_2\, y, \quad a_3 = u_3\, y$$

$$\gcd(u_1, u_2) = 1, \gcd(u_2, u_3) = 1$$

$$\frac{u_2}{u_1} = \cfrac{1}{m_1 + \cfrac{1}{m_2 + \cfrac{1}{m_3 + \cfrac{1}{\ddots + \cfrac{1}{m_r}}}}}, \qquad \frac{u_3}{u_2} = \cfrac{1}{m_2 + \cfrac{1}{m_3 + \cfrac{1}{\ddots + \cfrac{1}{m_r}}}},$$

$$\frac{u_2}{u_1} = h_{m_1} \circ h_{m_2} \circ \ldots \circ h_{m_r}(0), \qquad \frac{u_3}{u_2} = h_{m_2} \circ \ldots \circ h_{m_r}(0)$$

$$\frac{1}{u_1^2} = (h_{m_1} \circ h_{m_2} \circ \ldots \circ h_{m_r})'(0) \qquad \frac{1}{u_2^2} = (h_{m_2} \circ \ldots \circ h_{m_r})'(0)$$

Euclid Algorithm and Continued Fractions Expansions

We use Continued Fractions Expansions and LFT's $\quad h_m(x) := 1/(m+x)$

`Euclid`(a_1, a_2), [case $a_1 \geq a_2$].

$$y := \gcd(a_1, a_2); \qquad a_1 = u_1\, y, \quad a_2 = u_2\, y, \quad a_3 = u_3\, y$$

$$\gcd(u_1, u_2) = 1, \gcd(u_2, u_3) = 1$$

$$\frac{u_2}{u_1} = \cfrac{1}{m_1 + \cfrac{1}{m_2 + \cfrac{1}{m_3 + \cfrac{1}{\ddots + \cfrac{1}{m_r}}}}}, \qquad \frac{u_3}{u_2} = \cfrac{1}{m_2 + \cfrac{1}{m_3 + \cfrac{1}{\ddots + \cfrac{1}{m_r}}}},$$

$$\frac{u_2}{u_1} = h_{m_1} \circ h_{m_2} \circ \ldots \circ h_{m_r}(0), \qquad \frac{u_3}{u_2} = h_{m_2} \circ \ldots \circ h_{m_r}(0)$$

$$\frac{1}{u_1^2} = (h_{m_1} \circ h_{m_2} \circ \ldots \circ h_{m_r})'(0) \qquad \frac{1}{u_2^2} = (h_{m_2} \circ \ldots \circ h_{m_r})'(0)$$

For LFT's, the denominators are recovered with the derivatives

$$\text{if} \quad h(x) = \frac{ax+b}{cx+d}, \quad \text{then} \quad h'(x) = \frac{\det h}{(cx+d)^2}$$

Proof of the alternative expression for the Dirichlet generating function ($\ell = 2$)

$$\zeta(s_1)\zeta(s_2) = \zeta(s_1+s_2)\Big(\frac{1}{2}\ (I - \mathbf{G}_{s_1+s_2})^{-1} \circ (\mathbf{G}_{s_1} + \mathbf{G}_{s_2})\Big)[1](0)$$

with $\quad \mathbf{G}_s[f](t) = \displaystyle\sum_{h\in\mathcal{H}} |h'(t)|^{s/2} f\circ h(t) = \sum_{m\geq 1} \left(\frac{1}{m+t}\right)^s f\left(\frac{1}{m+t}\right)$

Proof of the alternative expression for the Dirichlet generating function ($\ell = 2$)

$$\zeta(s_1)\zeta(s_2) = \zeta(s_1+s_2)\left(\frac{1}{2}\,(I - \mathbf{G}_{s_1+s_2})^{-1} \circ (\mathbf{G}_{s_1} + \mathbf{G}_{s_2})\right)[1](0)$$

with $\quad \mathbf{G}_s[f](t) = \sum_{h\in\mathcal{H}} |h'(t)|^{s/2} f\circ h(t) = \sum_{m\geq 1} \left(\frac{1}{m+t}\right)^s f\left(\frac{1}{m+t}\right)$

Proof. The Dirichlet generating function of the inputs is

$$\sum_{(a_1,a_2)\in\mathbb{N}^2} \frac{1}{a_1^{s_1}}\frac{1}{a_2^{s_2}} = \zeta(s_1)\cdot\zeta(s_2), \qquad \zeta(s) = \sum_{a\in\mathbb{N}} \frac{1}{a^s}$$

Proof of the alternative expression for the Dirichlet generating function ($\ell = 2$)

$$\zeta(s_1)\zeta(s_2) = \zeta(s_1+s_2)\left(\frac{1}{2}\,(I - \mathbf{G}_{s_1+s_2})^{-1} \circ (\mathbf{G}_{s_1} + \mathbf{G}_{s_2})\right)[1](0)$$

with $\quad \mathbf{G}_s[f](t) = \sum_{h\in\mathcal{H}} |h'(t)|^{s/2} f\circ h(t) = \sum_{m\geq 1} \left(\frac{1}{m+t}\right)^s f\left(\frac{1}{m+t}\right)$

Proof. The Dirichlet generating function of the inputs is

$$\sum_{(a_1,a_2)\in\mathbb{N}^2} \frac{1}{a_1^{s_1}}\frac{1}{a_2^{s_2}} = \zeta(s_1)\cdot\zeta(s_2), \qquad \zeta(s) = \sum_{a\in\mathbb{N}} \frac{1}{a^s}$$

With the $\gcd y$: $\qquad \zeta(s_1)\zeta(s_2) = \left(\sum_{y\geq 1} \frac{1}{y^{s_1+s_2}}\right)\left(\sum_{\substack{(u_1,u_2)\in\mathbb{N}^2\\ \gcd(u_1,u_2)=1}} \frac{1}{u_1^{s_1}}\frac{1}{u_2^{s_2}}\right)$

Proof of the alternative expression for the Dirichlet generating function ($\ell = 2$)

$$\zeta(s_1)\zeta(s_2) = \zeta(s_1+s_2)\left(\frac{1}{2}\,(I - \mathbf{G}_{s_1+s_2})^{-1} \circ (\mathbf{G}_{s_1} + \mathbf{G}_{s_2})\right)[1](0)$$

with $\quad \mathbf{G}_s[f](t) = \sum_{h\in\mathcal{H}} |h'(t)|^{s/2} f\circ h(t) = \sum_{m\geq 1} \left(\frac{1}{m+t}\right)^s f\left(\frac{1}{m+t}\right)$

Proof. The Dirichlet generating function of the inputs is

$$\sum_{(a_1,a_2)\in\mathbb{N}^2} \frac{1}{a_1^{s_1}}\frac{1}{a_2^{s_2}} = \zeta(s_1)\cdot\zeta(s_2), \qquad \zeta(s) = \sum_{a\in\mathbb{N}} \frac{1}{a^s}$$

With the $\gcd y$: $\qquad \zeta(s_1)\zeta(s_2) = \left(\sum_{y\geq 1} \frac{1}{y^{s_1+s_2}}\right)\left(\sum_{\substack{(u_1,u_2)\in\mathbb{N}^2\\ \gcd(u_1,u_2)=1}} \frac{1}{u_1^{s_1}}\frac{1}{u_2^{s_2}}\right)$

Using LFT's as in the previous slide :

$$\sum_{\substack{(u_1,u_2),u_2\leq u_1\\ \gcd(u_1,u_2)=1}} \frac{1}{u_1^{s_1}}\frac{1}{u_2^{s_2}} = \sum_{\substack{g\in\mathcal{H},\\ h\in\mathcal{H}^\star}} |(g\circ h)'(0)|^{s_1/2}|h'(0)|^{s_2/2}$$

$$= \sum_{\substack{g\in\mathcal{H},\\ h\in\mathcal{H}^\star}} |g'(h(0))|^{s_1/2}|h'(0)|^{(s_1+s_2)/2}$$

Proof of the alternative expression for the Dirichlet generating function ($\ell = 2$)

$$\zeta(s_1)\zeta(s_2) = \zeta(s_1+s_2)\left(\frac{1}{2}\,(I - \mathbf{G}_{s_1+s_2})^{-1} \circ (\mathbf{G}_{s_1} + \mathbf{G}_{s_2})\right)[1](0)$$

with $\quad \mathbf{G}_s[f](t) = \sum_{h\in\mathcal{H}} |h'(t)|^{s/2} f\circ h(t) = \sum_{m\geq 1} \left(\frac{1}{m+t}\right)^s f\left(\frac{1}{m+t}\right)$

Proof. The Dirichlet generating function of the inputs is

$$\sum_{(a_1,a_2)\in\mathbb{N}^2} \frac{1}{a_1^{s_1}}\frac{1}{a_2^{s_2}} = \zeta(s_1)\cdot\zeta(s_2), \qquad \zeta(s) = \sum_{a\in\mathbb{N}} \frac{1}{a^s}$$

With the $\gcd y$: $\qquad \zeta(s_1)\zeta(s_2) = \left(\sum_{y\geq 1} \frac{1}{y^{s_1+s_2}}\right)\left(\sum_{\substack{(u_1,u_2)\in\mathbb{N}^2\\ \gcd(u_1,u_2)=1}} \frac{1}{u_1^{s_1}}\frac{1}{u_2^{s_2}}\right)$

Using LFT's as in the previous slide :

$$\sum_{\substack{(u_1,u_2),u_2\leq u_1\\ \gcd(u_1,u_2)=1}} \frac{1}{u_1^{s_1}}\frac{1}{u_2^{s_2}} = \sum_{\substack{g\in\mathcal{H},\\ h\in\mathcal{H}^\star}} |(g\circ h)'(0)|^{s_1/2}|h'(0)|^{s_2/2}$$

$$= \sum_{\substack{g\in\mathcal{H},\\ h\in\mathcal{H}^\star}} |g'(h(0))|^{s_1/2}|h'(0)|^{(s_1+s_2)/2} = (I - \mathbf{G}_{s_1+s_2})^{-1}\circ \mathbf{G}_{s_1}[1](0)$$

The Euclid algorithm ($\ell = 2$) on integers

translates as a product of Dirichlet generating functions

$$\zeta(s_1)\,\zeta(s_2) = T(s_1, s_2)\;\zeta(s_1 + s_2),$$

with $$T(s_1, s_2) = \frac{1}{2}(1 - \mathbf{G}_{s_1+s_2})^{-1} \circ (\mathbf{G}_{s_1} + \mathbf{G}_{s_2})\,[1](0)$$

The Euclid algorithm ($\ell = 2$) on integers

translates as a product of Dirichlet generating functions

$$\zeta(s_1)\,\zeta(s_2) = T(s_1, s_2)\ \zeta(s_1 + s_2),$$

with $\quad T(s_1, s_2) = \dfrac{1}{2}(1 - \mathbf{G}_{s_1+s_2})^{-1} \circ (\mathbf{G}_{s_1} + \mathbf{G}_{s_2})\,[1](0)$

Then, for any $\ell \geq 2$, the ℓ–Euclid algorithm translates as the product

$$\zeta(s_1) \cdot \ldots \cdot \zeta(s_\ell) = \zeta(t_\ell) \prod_{k=1}^{\ell-1} T(s_{k+1}, t_k) \qquad [t_k := s_1 + s_2 + \ldots + s_k,]$$

The Euclid algorithm ($\ell = 2$) on integers

translates as a product of Dirichlet generating functions

$$\zeta(s_1)\,\zeta(s_2) = T(s_1, s_2)\;\zeta(s_1 + s_2),$$

with $\quad T(s_1, s_2) = \dfrac{1}{2}(1 - \mathbf{G}_{s_1+s_2})^{-1} \circ (\mathbf{G}_{s_1} + \mathbf{G}_{s_2})\,[1](0)$

Then, for any $\ell \geq 2$, the ℓ–Euclid algorithm translates as the product

$$\zeta(s_1) \cdot \ldots \cdot \zeta(s_\ell) = \zeta(t_\ell) \prod_{k=1}^{\ell-1} T(s_{k+1}, t_k) \qquad [t_k := s_1 + s_2 + \ldots + s_k,]$$

Now, with $s = s_1 = \ldots = s_\ell$,
the (plain) generating function $S(s)$ of $\mathbb{N}^\ell$ has an alternative expression

$$S(s) = \zeta(s)^\ell = \zeta(\ell s) \prod_{k=1}^{\ell-1} T(s, ks)$$

which is an exact translation of the ℓ-Euclid algorithm.

T is the "phase generating function".

Bivariate Generating functions for the ℓ-Euclid Algorithm

We start with: $$S(s) = \zeta(s)^\ell = \zeta(\ell s) \prod_{k=1}^{\ell-1} T(s, ks)$$

Bivariate Generating functions for the ℓ-Euclid Algorithm

We start with: $$S(s) = \zeta(s)^\ell = \zeta(\ell s) \prod_{k=1}^{\ell-1} T(s, ks)$$

For studying the distribution of the two costs :

- L_k (number of steps in the k-th phase)
- D_k (size of the gcd at the beginning of the k-th phase)

we use bivariate generating functions, with an extra variable u

Bivariate Generating functions for the ℓ-Euclid Algorithm

We start with: $$S(s) = \zeta(s)^\ell = \zeta(\ell s) \prod_{k=1}^{\ell-1} T(s, ks)$$

For studying the distribution of the two costs :

- L_k (number of steps in the k-th phase)
- D_k (size of the gcd at the beginning of the k-th phase)

we use bivariate generating functions, with an extra variable u

$$L_k(s,u) = \zeta(s)^\ell \cdot \frac{T(s,ks,u)}{T(s,ks)}, \qquad D_k(s,u) = \zeta(s)^\ell \cdot \frac{Z(ks,u)}{\zeta(ks)},$$

$$T(s,t,u) = \frac{1}{2} u (1 - u\mathbf{G}_{s+t})^{-1} \circ (\mathbf{G}_s + \mathbf{G}_t)[1](0), \qquad Z(s,u) = \sum_{n \geq 1} \frac{u^{\mathrm{d}(n)}}{n^s}$$

Towards the distributional analysis of L_k and D_k.

The generating functions of the events $[L_k > m]$ and $[D_k > m]$

$$\widehat{L}_k^{[m]}(s) := \sum_{j>m} [u^j] L_k(s,u), \qquad \widehat{D}_k^{[m]}(s) = \sum_{j>m} [u^j] D_k(s,u)$$

Towards the distributional analysis of L_k and D_k.

The generating functions of the events $[L_k > m]$ and $[D_k > m]$

$$\widehat{L}_k^{[m]}(s) := \sum_{j>m} [u^j] L_k(s,u), \qquad \widehat{D}_k^{[m]}(s) = \sum_{j>m} [u^j] D_k(s,u)$$

admit the alternative expressions:

with $\zeta_M(s) := \sum_{n \geq M} \frac{1}{n^s}$ and $\varphi_{s,t}$ some "nice" function,

$$\widehat{D}_k^{[m]}(s) = \zeta(s)^\ell \cdot \frac{\zeta_{e^m}(ks)}{\zeta(ks)} \qquad \widehat{L}_k^{[m]}(s) = \zeta(s)^\ell \cdot \mathbf{G}^m_{(k+1)s}[\varphi_{ks,s}](0)$$

Towards the distributional analysis of L_k and D_k.

The generating functions of the events $[L_k > m]$ and $[D_k > m]$

$$\widehat{L}_k^{[m]}(s) := \sum_{j>m} [u^j] L_k(s,u), \qquad \widehat{D}_k^{[m]}(s) = \sum_{j>m} [u^j] D_k(s,u)$$

admit the alternative expressions:

with $\zeta_M(s) := \sum_{n \geq M} \frac{1}{n^s}$ and $\varphi_{s,t}$ some "nice" function,

$$\widehat{D}_k^{[m]}(s) = \zeta(s)^\ell \cdot \frac{\zeta_{e^m}(ks)}{\zeta(ks)} \qquad \widehat{L}_k^{[m]}(s) = \zeta(s)^\ell \cdot \mathbf{G}^m_{(k+1)s}[\varphi_{ks,s}](0)$$

both of type $B(s) \cdot A_{k,m}(s)$, with $A_{k,m}(s) \approx A_k^m(s)$

$A_k(s) = \lambda((k+1)s)$ [L–case] $\qquad A_k(s) = \exp[1-ks]$ [D–case]

Towards the distributional analysis of L_k and D_k.

The generating functions of the events $[L_k > m]$ and $[D_k > m]$

$$\widehat{L}_k^{[m]}(s) := \sum_{j>m} [u^j] L_k(s,u), \qquad \widehat{D}_k^{[m]}(s) = \sum_{j>m} [u^j] D_k(s,u)$$

admit the alternative expressions:

with $\zeta_M(s) := \sum_{n \geq M} \frac{1}{n^s}$ and $\varphi_{s,t}$ some "nice" function,

$$\widehat{D}_k^{[m]}(s) = \zeta(s)^\ell \cdot \frac{\zeta_{e^m}(ks)}{\zeta(ks)} \qquad \widehat{L}_k^{[m]}(s) = \zeta(s)^\ell \cdot \mathbf{G}^m_{(k+1)s}[\varphi_{ks,s}](0)$$

both of type $B(s) \cdot A_{k,m}(s)$, with $A_{k,m}(s) \approx A_k^m(s)$

$$A_k(s) = \lambda((k+1)s) \quad [L\text{–case}] \qquad A_k(s) = \exp[1-ks] \quad [D\text{–case}]$$

The asymptotics depends on the value $a := A_k(1)$ at the pole $s = 1$ of $B(s)$

For $k = 1$, one has $a = 1$ – For $k \geq 2$, one has $a < 1$.

Main result for the number of divisions L_k – First phase ($k = 1$)

Main result for the number of divisions L_k – First phase ($k = 1$)

The number of divisions L_1 performed during the first phase

– has a mean value of linear order

$$\mathbb{E}_n[L_1] = \frac{6\log 2}{\pi^2}\,\frac{n}{\ell}\left[1 + O\left(\frac{1}{n}\right)\right] \qquad \frac{\pi^2}{6\log 2} = \text{entropy}$$

Main result for the number of divisions L_k – First phase $(k = 1)$

The number of divisions L_1 performed during the first phase

– has a mean value of linear order

$$\mathbb{E}_n[L_1] = \frac{6\log 2}{\pi^2}\,\frac{n}{\ell}\left[1 + O\left(\frac{1}{n}\right)\right] \qquad \frac{\pi^2}{6\log 2} = \text{entropy}$$

– follows an asymptotic beta law of parameter $(1, \ell - 1)$.

Its distribution satisfies when $n \to \infty$, and $m/n \in [0, (6\log 2)/\pi^2]$

$$\mathbb{P}[L_1 > m] = \left(1 - \frac{m}{n}\,\frac{\pi^2}{6\log 2}\right)^{\ell-1} + O\left(\frac{1}{n^\alpha}\right)$$

Main result for the number of divisions L_k – First phase $(k = 1)$

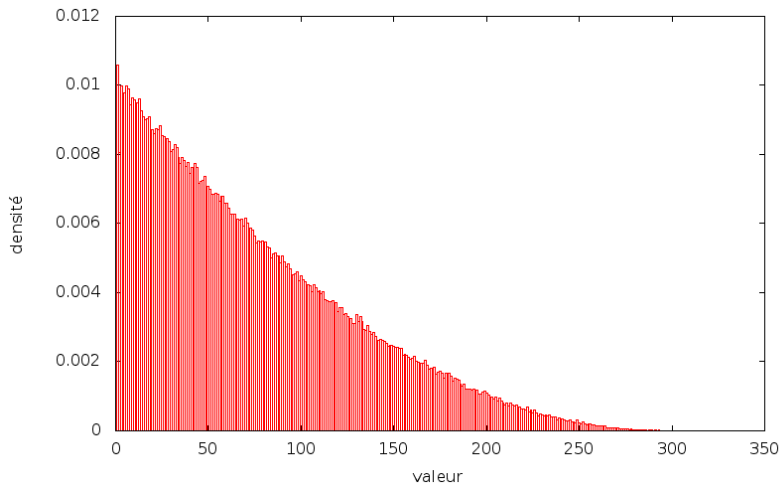
The number of divisions L_1 performed during the first phase

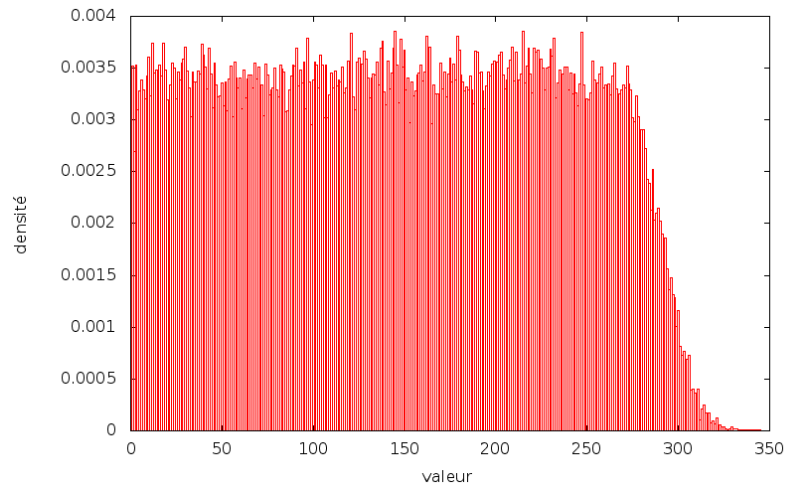
– has a mean value of linear order

$$\mathbb{E}_n[L_1] = \frac{6\log 2}{\pi^2}\,\frac{n}{\ell}\left[1 + O\left(\frac{1}{n}\right)\right] \qquad \frac{\pi^2}{6\log 2} = \text{entropy}$$

– follows an asymptotic beta law of parameter $(1, \ell - 1)$.
Its distribution satisfies when $n \to \infty$, and $m/n \in [0, (6\log 2)/\pi^2]$

$$\mathbb{P}[L_1 > m] = \left(1 - \frac{m}{n}\,\frac{\pi^2}{6\log 2}\right)^{\ell-1} + O\left(\frac{1}{n^\alpha}\right)$$





Main result for the number of divisions L_k – Subsequent phases (case $k \geq 2$)

For $k \geq 2$, the number of divisions during the k-th phase

Main result for the number of divisions L_k – Subsequent phases (case $k \geq 2$)

For $k \geq 2$, the number of divisions during the k-th phase

– has a mean value of constant order

$$\mathbb{E}_n[L_k] = (I - \mathbf{G}_{k+1})^{-1}[\varphi_{k,1}](0) + O\left(\frac{1}{n}\right)$$

Main result for the number of divisions L_k – Subsequent phases (case $k \geq 2$)

For $k \geq 2$, the number of divisions during the k-th phase

– has a mean value of constant order

$$\mathbb{E}_n[L_k] = (I - \mathbf{G}_{k+1})^{-1}[\varphi_{k,1}](0) + O\left(\frac{1}{n}\right)$$

– follows an asymptotic quasi-geometric law, with quasi–ratio $\lambda(k+1)$

For $n \to \infty$ and $m/n \in [0, |\lambda(k+1)/\lambda'(k+1)|]$,

$$\mathbb{P}_n[L_k > m] = \mathbf{G}_{k+1}^m[\varphi_{k,1}](0) + O\left(\frac{\log n}{n}\right)$$

Main result for the number of divisions L_k – Subsequent phases (case $k \geq 2$)

For $k \geq 2$, the number of divisions during the k-th phase

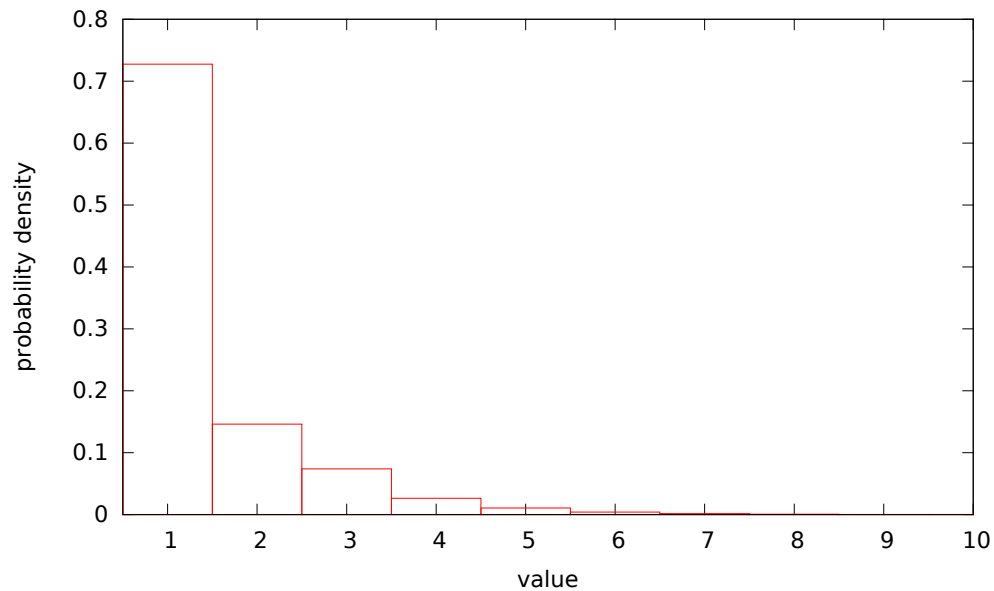
– has a mean value of constant order

$$\mathbb{E}_n[L_k] = (I - \mathbf{G}_{k+1})^{-1}[\varphi_{k,1}](0) + O\left(\frac{1}{n}\right)$$

– follows an asymptotic quasi-geometric law, with quasi–ratio $\lambda(k+1)$

For $n \to \infty$ and $m/n \in [0, |\lambda(k+1)/\lambda'(k+1)|]$,

$$\mathbb{P}_n[L_k > m] = \mathbf{G}_{k+1}^m[\varphi_{k,1}](0) + O\left(\frac{\log n}{n}\right)$$



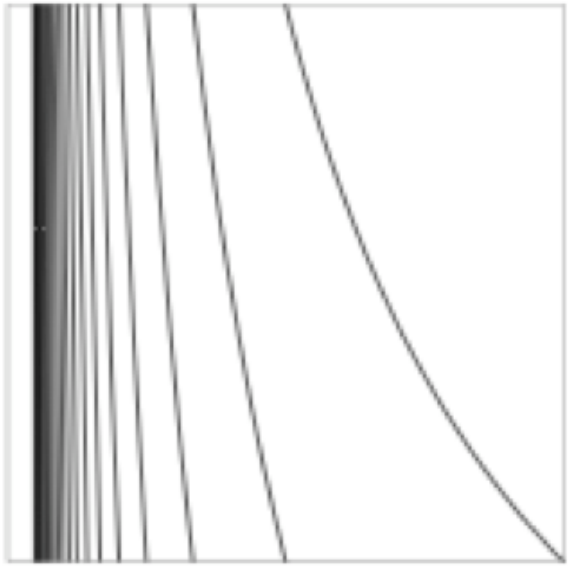
IV. An unified point of view for the two analyses?

A dynamical point of view....

The underlying dynamical system in the integer case

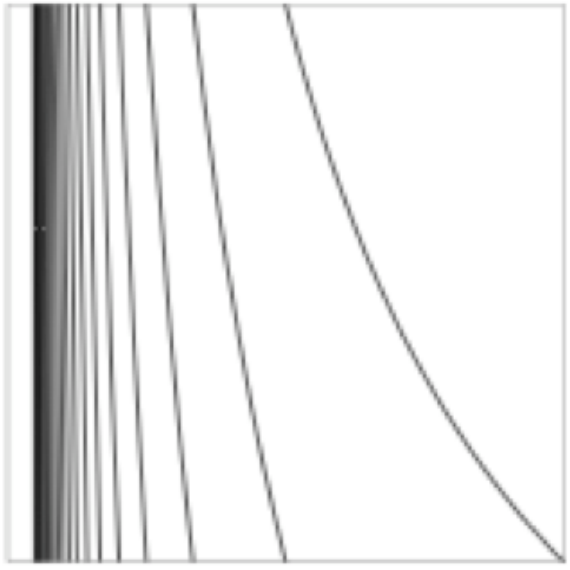
The underlying dynamical system in the integer case defined by the pair $([0,1], V)$.

$$V : [0,1] \longrightarrow [0,1], \quad V(x) := \frac{1}{x} - \left\lfloor \frac{1}{x} \right\rfloor \quad \text{for} \quad x \neq 0, \quad V(0) = 0$$



The underlying dynamical system in the integer case defined by the pair $([0,1], V)$.

$$V : [0,1] \longrightarrow [0,1], \quad V(x) := \frac{1}{x} - \left\lfloor \frac{1}{x} \right\rfloor \quad \text{for} \quad x \neq 0, \quad V(0) = 0$$



The set of the inverse branches of V is

$$\left\{ h_m(x) = \frac{1}{m+x} \right\}$$

It builds the continued fraction expansions.

The operator $\mathbf{G}_s$ is the transfer operator of the dynamical system.
It involves the derivatives of the inverse branches of V

$$\mathbf{G}_s[f](x) = \sum_{m \geq 1} \left(\frac{1}{m+x} \right)^s f\left(\frac{1}{m+x} \right)$$

The underlying dynamical system in the polynomial case.

The underlying dynamical system in the polynomial case.

The exact ultrametric analog of the DS in the integer case.

$$\mathbb{F}_q[Z] \longleftrightarrow \mathbb{Z}, \qquad \mathbb{F}_q(Z) \longleftrightarrow \mathbb{Q}, \qquad \mathbb{F}_q((1/Z)) \longleftrightarrow \mathbb{R}$$

$\mathbb{F}_q((1/Z))$: the completion of $\mathbb{F}_q[Z]$ for the ultrametric norm $||u|| := q^{\mathrm{d}(u)}$.

The integer part function $\lfloor \cdot \rfloor$ satisfies $\lfloor f \rfloor = 0$ iff $||f|| \leq 1$

The underlying dynamical system in the polynomial case.

The exact ultrametric analog of the DS in the integer case.

$$\mathbb{F}_q[Z] \longleftrightarrow \mathbb{Z}, \qquad \mathbb{F}_q(Z) \longleftrightarrow \mathbb{Q}, \qquad \mathbb{F}_q((1/Z)) \longleftrightarrow \mathbb{R}$$

$\mathbb{F}_q((1/Z))$: the completion of $\mathbb{F}_q[Z]$ for the ultrametric norm $||u|| := q^{\mathrm{d}(u)}$.

The integer part function $\lfloor \cdot \rfloor$ satisfies $\lfloor f \rfloor = 0$ iff $||f|| \leq 1$

Dynamical system: $(\mathcal{X}_q, V)$ with

$$\mathcal{X}_q := \{f \in \mathbb{F}_q((1/Z)); \quad ||f|| \leq 1\} \text{ and}$$

$$V : \mathcal{X}_q \to \mathcal{X}_q \qquad V(x) = \frac{1}{x} - \left\lfloor \frac{1}{x} \right\rfloor \quad (x \neq 0), \qquad V(0) = 0.$$

The underlying dynamical system in the polynomial case.

The exact ultrametric analog of the DS in the integer case.

$$\mathbb{F}_q[Z] \longleftrightarrow \mathbb{Z}, \qquad \mathbb{F}_q(Z) \longleftrightarrow \mathbb{Q}, \qquad \mathbb{F}_q((1/Z)) \longleftrightarrow \mathbb{R}$$

$\mathbb{F}_q((1/Z))$: the completion of $\mathbb{F}_q[Z]$ for the ultrametric norm $||u|| := q^{\mathrm{d}(u)}$.

The integer part function $\lfloor \cdot \rfloor$ satisfies $\lfloor f \rfloor = 0$ iff $||f|| \leq 1$

Dynamical system: $(\mathcal{X}_q, V)$ with

$$\mathcal{X}_q := \{f \in \mathbb{F}_q((1/Z)); \quad ||f|| \leq 1\} \text{ and}$$

$$V : \mathcal{X}_q \to \mathcal{X}_q \qquad V(x) = \frac{1}{x} - \left\lfloor \frac{1}{x} \right\rfloor \quad (x \neq 0), \qquad V(0) = 0.$$

The transfer operator $\mathbf{G}_s$ is defined with the inverse branches of V,

$$\mathbf{G}_s[f](x) = \sum_{m, ||m|| \geq 1} \frac{1}{\|m + x\|^s} \, f\left(\frac{1}{m+x}\right)$$

The underlying dynamical system in the polynomial case.

The exact ultrametric analog of the DS in the integer case.

$$\mathbb{F}_q[Z] \longleftrightarrow \mathbb{Z}, \qquad \mathbb{F}_q(Z) \longleftrightarrow \mathbb{Q}, \qquad \mathbb{F}_q((1/Z)) \longleftrightarrow \mathbb{R}$$

$\mathbb{F}_q((1/Z))$: the completion of $\mathbb{F}_q[Z]$ for the ultrametric norm $||u|| := q^{\mathrm{d}(u)}$.

The integer part function $\lfloor \cdot \rfloor$ satisfies $\lfloor f \rfloor = 0$ iff $||f|| \leq 1$

Dynamical system: $(\mathcal{X}_q, V)$ with

$$\mathcal{X}_q := \{f \in \mathbb{F}_q((1/Z)); \quad ||f|| \leq 1\} \text{ and}$$

$$V : \mathcal{X}_q \to \mathcal{X}_q \qquad V(x) = \frac{1}{x} - \left\lfloor \frac{1}{x} \right\rfloor \quad (x \neq 0), \qquad V(0) = 0.$$

The transfer operator $\mathbf{G}_s$ is defined with the inverse branches of V,

$$\mathbf{G}_s[f](x) = \sum_{m, ||m|| \geq 1} \frac{1}{\|m + x\|^s} \, f\left(\frac{1}{m+x}\right)$$

$$\text{Ultrametric norm} \Longrightarrow ||m + x|| = ||m|| \Longrightarrow \mathbf{G}_s[1] = \sum_{m, ||m|| > 1} \frac{1}{||m||^s}$$

$$\mathbf{G}_s[1] = \sum_{m, \mathrm{d}(m) > 0} \frac{1}{q^{\mathrm{d}(m)s}} = \text{the power generating function } G(z) \text{ with } z = q^{-s}$$

The underlying dynamical system in the polynomial case.

The exact ultrametric analog of the DS in the integer case.

$$\mathbb{F}_q[Z] \longleftrightarrow \mathbb{Z}, \qquad \mathbb{F}_q(Z) \longleftrightarrow \mathbb{Q}, \qquad \mathbb{F}_q((1/Z)) \longleftrightarrow \mathbb{R}$$

$\mathbb{F}_q((1/Z))$: the completion of $\mathbb{F}_q[Z]$ for the ultrametric norm $||u|| := q^{\mathrm{d}(u)}$.

The integer part function $\lfloor \cdot \rfloor$ satisfies $\lfloor f \rfloor = 0$ iff $||f|| \leq 1$

Dynamical system: $(\mathcal{X}_q, V)$ with

$$\mathcal{X}_q := \{f \in \mathbb{F}_q((1/Z)); \quad ||f|| \leq 1\} \text{ and}$$

$$V : \mathcal{X}_q \to \mathcal{X}_q \qquad V(x) = \frac{1}{x} - \left\lfloor \frac{1}{x} \right\rfloor \quad (x \neq 0), \qquad V(0) = 0.$$

The transfer operator $\mathbf{G}_s$ is defined with the inverse branches of V,

$$\mathbf{G}_s[f](x) = \sum_{m, ||m|| \geq 1} \frac{1}{\|m + x\|^s} \, f\left(\frac{1}{m+x}\right)$$

$$\text{Ultrametric norm} \Longrightarrow ||m + x|| = ||m|| \Longrightarrow \mathbf{G}_s[1] = \sum_{m, ||m|| > 1} \frac{1}{||m||^s}$$

$$\mathbf{G}_s[1] = \sum_{m, \mathrm{d}(m) > 0} \frac{1}{q^{\mathrm{d}(m)s}} = \text{the power generating function } G(z) \text{ with } z = q^{-s}$$

Classical analysis for polynomials: a particular case of a dynamical analysis

General conclusions.

Methodologic point of view.

– A simple (but not trivial) instance of analysis of algorithms
  – Classical analysis versus dynamical analysis
  – Provides precise distributional results.
  – With an (unexpected) occurrence of the Beta law

General conclusions.

Methodologic point of view.

– A simple (but not trivial) instance of analysis of algorithms
  – Classical analysis versus dynamical analysis
  – Provides precise distributional results.
  – With an (unexpected) occurrence of the Beta law
– Can be extended to the analysis of the bit complexity [work in progress]

General conclusions.

Methodologic point of view.

– A simple (but not trivial) instance of analysis of algorithms
  – Classical analysis versus dynamical analysis
  – Provides precise distributional results.
  – With an (unexpected) occurrence of the Beta law
– Can be extended to the analysis of the bit complexity [work in progress]

Algorithmic point of view.

– This plain strategy is proven to be efficient.
– To be compared to other strategies

General conclusions.

Methodologic point of view.

– A simple (but not trivial) instance of analysis of algorithms
  – Classical analysis versus dynamical analysis
  – Provides precise distributional results.
  – With an (unexpected) occurrence of the Beta law
– Can be extended to the analysis of the bit complexity [work in progress]

Algorithmic point of view.

– This plain strategy is proven to be efficient.
– To be compared to other strategies
  – Random scalar products [Von Zur Gathen, Shparlinski],
    Compute a unique gcd between two random scalar products

General conclusions.

Methodologic point of view.

– A simple (but not trivial) instance of analysis of algorithms
  – Classical analysis versus dynamical analysis
  – Provides precise distributional results.
  – With an (unexpected) occurrence of the Beta law
– Can be extended to the analysis of the bit complexity [work in progress]

Algorithmic point of view.

– This plain strategy is proven to be efficient.
– To be compared to other strategies
  – Random scalar products [Von Zur Gathen, Shparlinski],
    Compute a unique gcd between two random scalar products
  – Brun's algorithm : [Joint work in progress with Berthé and Lhote]