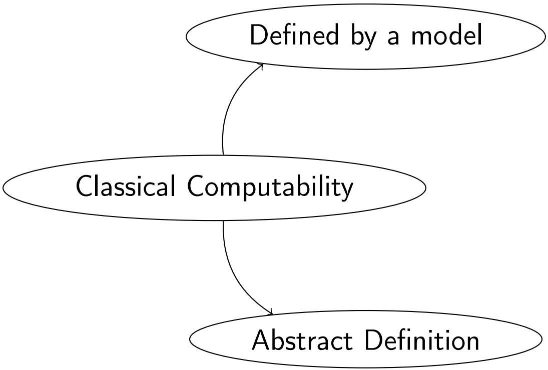
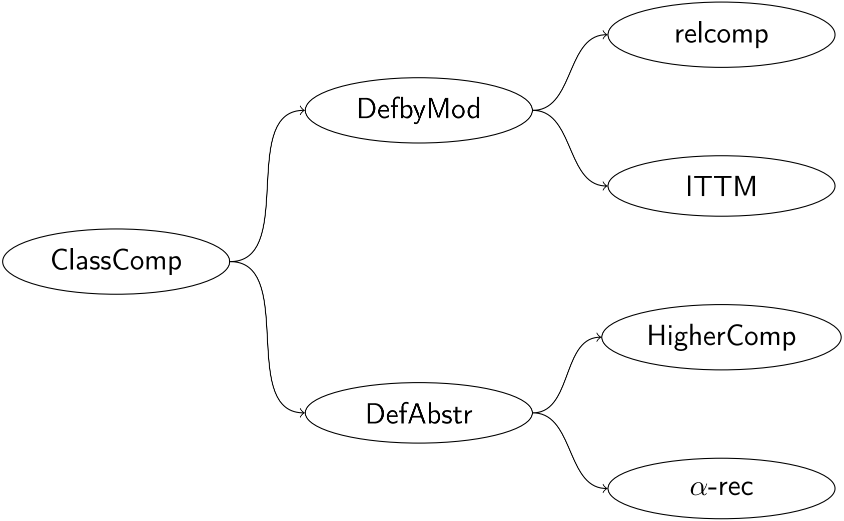
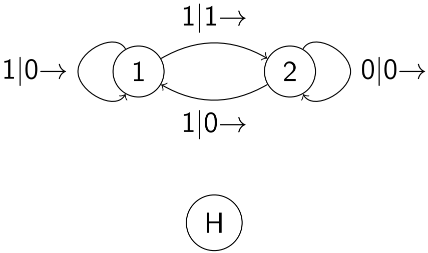
Higher Computability and Randomness

Paul-Elliot Anglès d'Auriac
Benoît Monin

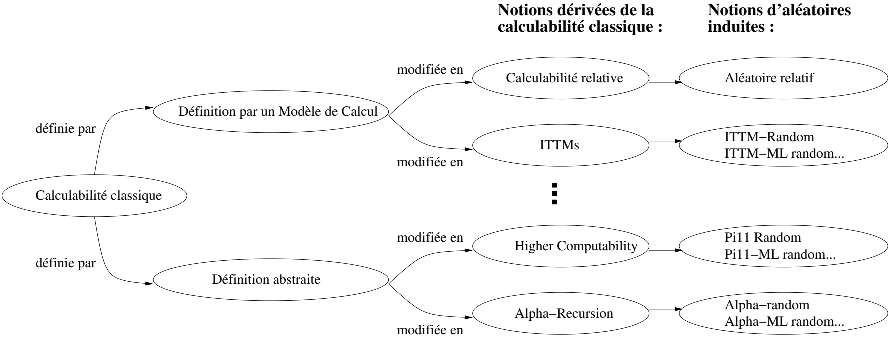
10 mai 2017


Defined by a model
Classical Computability
Abstract Definition

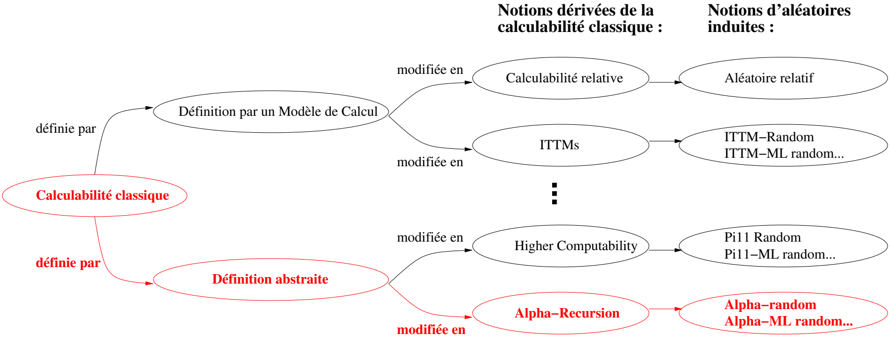

ClassComp
DefbyMod
DefAbstr
relcomp
ITTM
HigherComp
α-rec


1|1→
1|0→
1
2
0|0→
1|0→
H

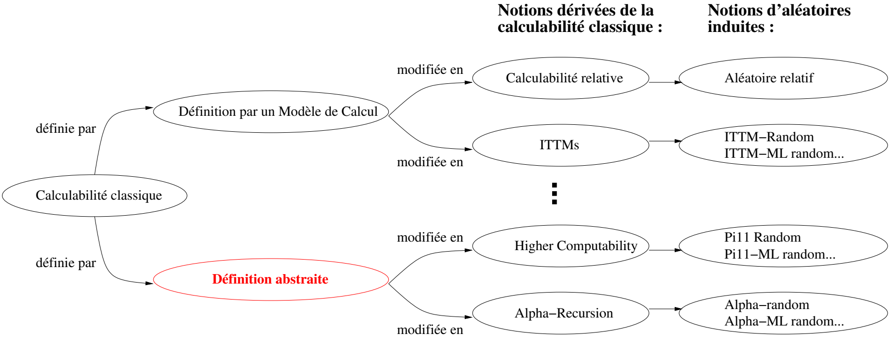
Étendre la calculabilité



Plan de l'exposé



Première étape

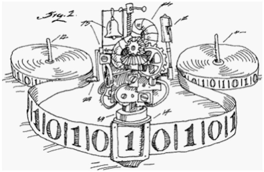


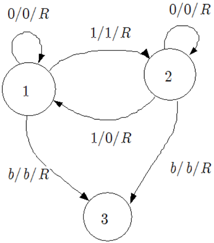
Infinite Time Turing Machine

Définition (Turing Machine)

A Turing Machine is an abstract model of computation, with :

- a finite quantity of states,
- an inifinite tape as memory, with a reading head,
- some transitions between states.



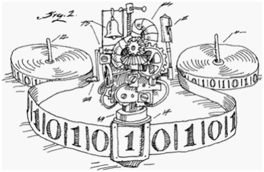


Infinite Time Turing Machine

Définition (Turing Machine)

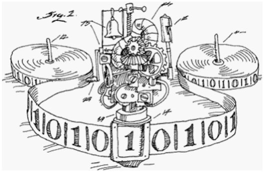
A Turing Machine is an abstract model of computation, with :

- a finite quantity of states,
- an inifinite tape as memory, with a reading head,
- some transitions between states.



```
10 Input A
15 Input B
20 B = A + 10
30 IF B > 12 GOTO 60
40 C = B / 3
50 IF C < 24 GOTO 10
60 Write C
70 IF Write Failed GOTO 15
80 Input D
```

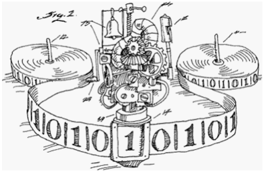
A computation



in state q_0.

A Turing Machine's Clock :

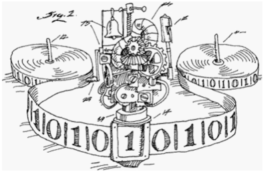
A computation



in state q_1.

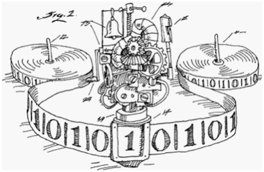
A Turing Machine's Clock :

A computation

 in state q_2.

A Turing Machine's Clock :

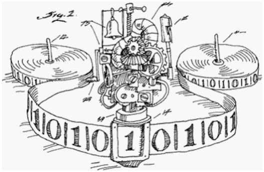
A computation



in state q_3.

A Turing Machine's Clock :

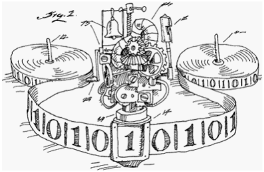
A computation



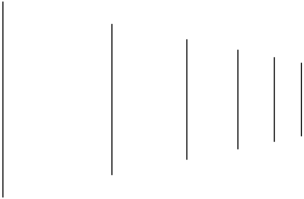
in state q_4.

A Turing Machine's Clock :

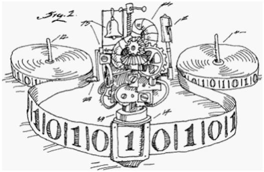
A computation

 in state q_5.

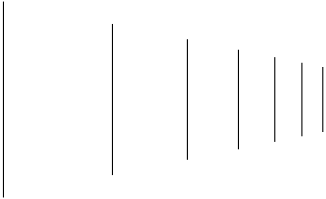
A Turing Machine's Clock :



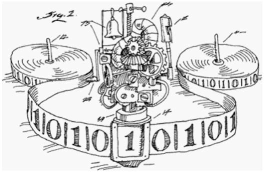
A computation

 in state q_6.

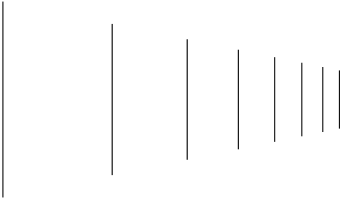
A Turing Machine's Clock :



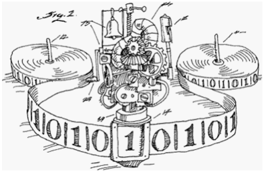
A computation

 in state q_7.

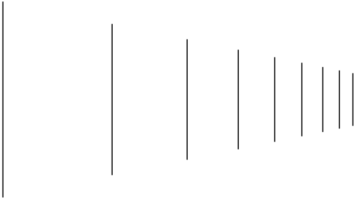
A Turing Machine's Clock :



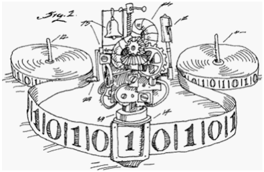
A computation

 in state q_8.

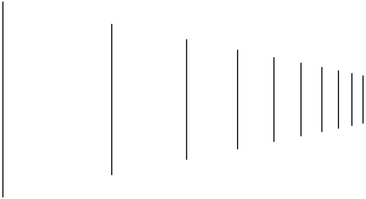
A Turing Machine's Clock :



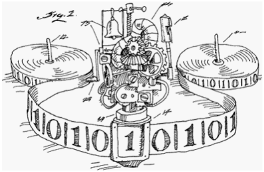
A computation

 in state q_9.

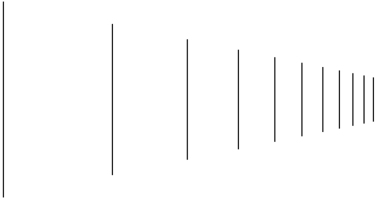
A Turing Machine's Clock :



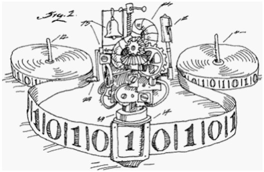
A computation

 in state q_{10}.

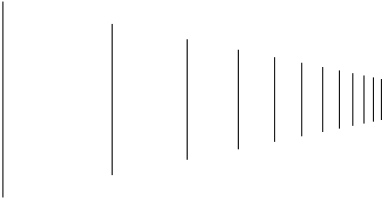
A Turing Machine's Clock :



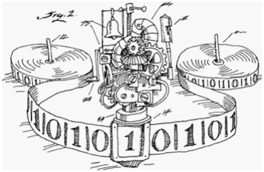
A computation

 in state q_{11}.

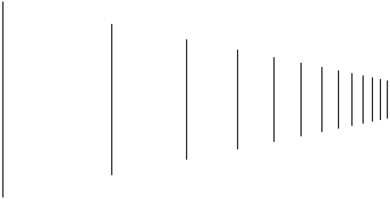
A Turing Machine's Clock :



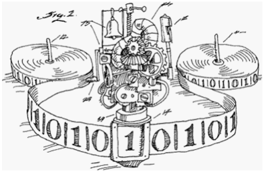
A computation

 in state q_{12}.

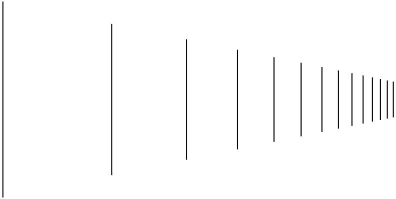
A Turing Machine's Clock :



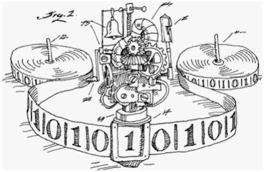
A computation

 in state q_{13}.

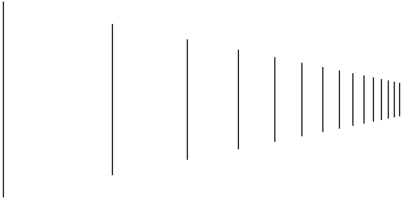
A Turing Machine's Clock :



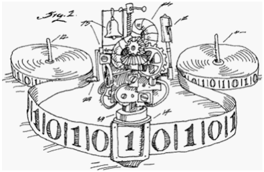
A computation

 in state q_{14}.

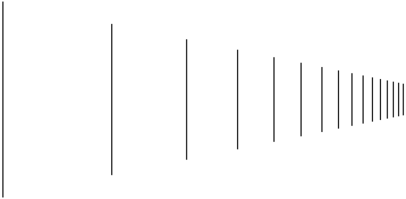
A Turing Machine's Clock :



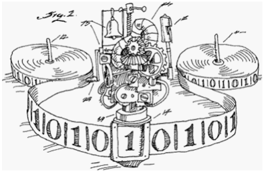
A computation

 in state q_{15}.

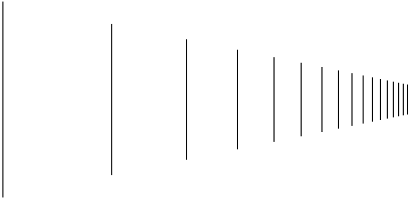
A Turing Machine's Clock :



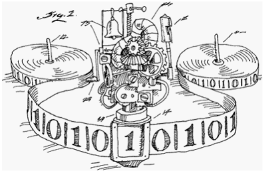
A computation

 in state q_{16}.

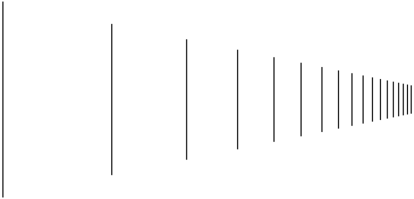
A Turing Machine's Clock :



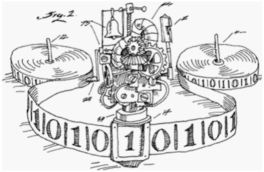
A computation

 in state q_{17}.

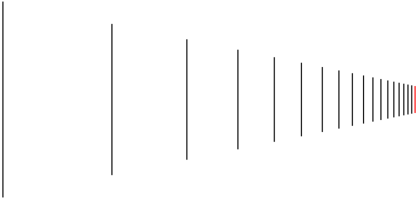
A Turing Machine's Clock :



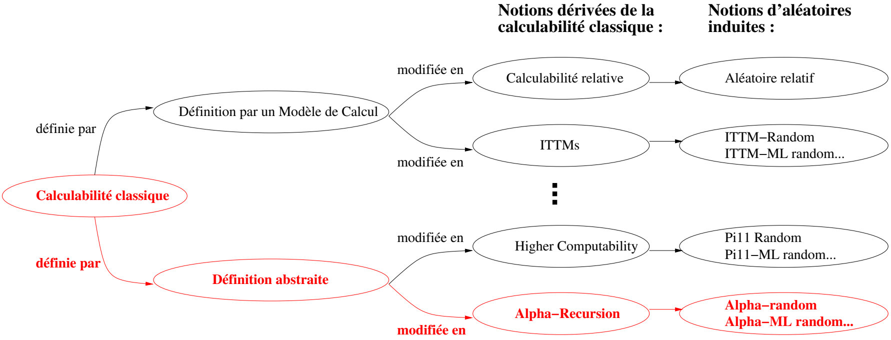
A computation


in state H.

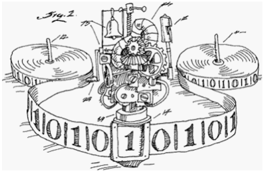
A Turing Machine's Clock :



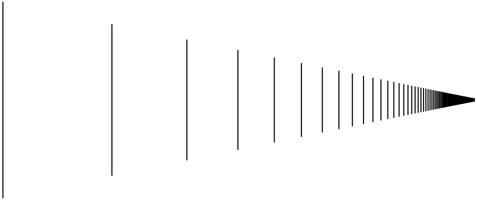
Plan de l'exposé



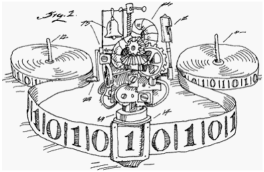
Hardware Clock



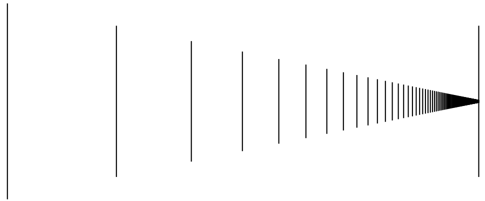
The Hardware Clock of all Turing Machines (compactified) :



New Hardware Clock



What if we consider instead :



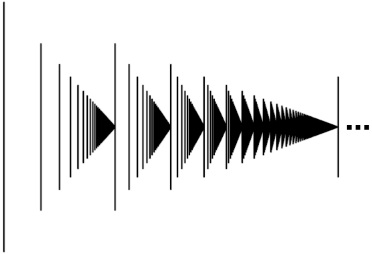
or even more ticks ?

Ordinals

Définition

An ordinal is a set α such that

1. α is transitive : $\forall x \in \alpha, \forall y \in x, y \in \alpha$
2. $(\alpha, \in)$ is a well ordering.



- Some ordinals are successors,
- some ordinals are limits.

Behaviour

Question

What is the behaviour at a limit stage ?

Answer

- The current state become the "limit" state,
- each cell become the liminf of its previous values.

Cellule 0 was : 1,0,0,1,1,1,1,1,1,1,1,... $\rightarrow$ now it is 1.
Cellule 1 was : 1,1,1,0,1,0,1,1,0,0,1,... $\rightarrow$ now it is 0.

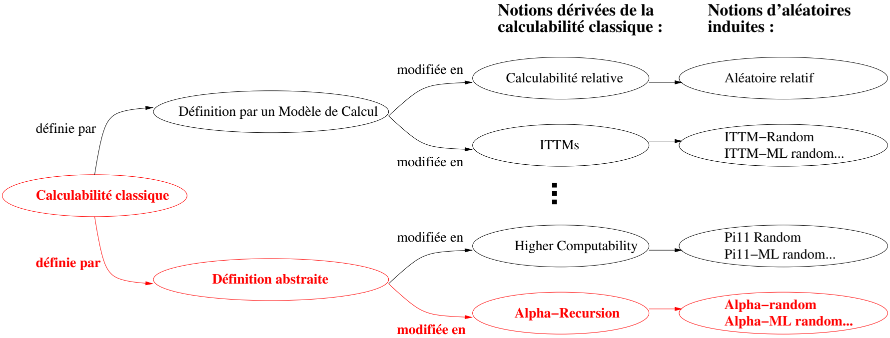
Links between Définability and Computability

- Is the behaviour original as far as we go through the ordinals ?
- What can these machines compute ?
- What's all that for ?

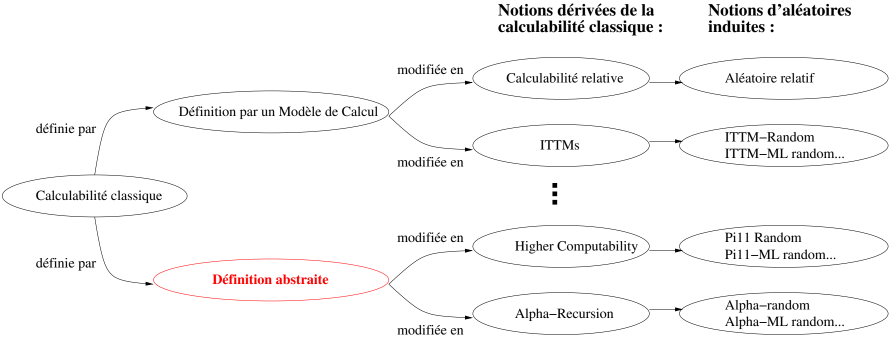
Links between Définability and Computability

- Is the behaviour original as far as we go through the ordinals ?
  - No, no machine would stop if it reach a certain ordinal, λ, no machine would stabilize if it reach another ordinal, ζ, and all machine will loop if it reaches a third ordinal ordinal, Σ.
- What can these machines compute ?
  - For example, the Halting Problem can be written on a tape ;
  - however, only countably many strings are writable ;
  - we will see soon a caracterization of what can be computed.
- What's all that for ?
  - The ordinals λ, ζ and Σ have interesting properties.
  - A new notion helps us understand the previous one.

Plan de l'exposé



Plan de l'exposé



Abstract ourselves from computationnal model

Denote HF the set consisting of all hereditarily finite sets. The following theorem caracterise the notion of "being computable" :

Théorème

Let $A \subseteq \mathbb{N}$, then :

1. *A is computable iff A is Δ_1-comprehensible in HF,*
2. *A is recursively enumerable iff A is Σ_1-comprehensible in HF,*

Abstract ourselves from computationnal model

Denote HF the set consisting of all hereditarily finite sets. The following theorem caracterise the notion of "being computable" :

Théorème

Let $A \subseteq \mathbb{N}$, then :

1. *A is computable iff A is Δ_1-comprehensible in HF,*
2. *A is recursively enumerable iff A is Σ_1-comprehensible in HF,*

1. Can be extended to $A \subseteq \mathrm{HF}$;
2. Can be modified by replacing HF by a well chosen set.

Abstract ourselves from computationnal model

Denote HF the set consisting of all hereditarily finite sets. The following theorem caracterise the notion of "being computable" :

Théorème

Let $A \subseteq \mathrm{HF}$, then :

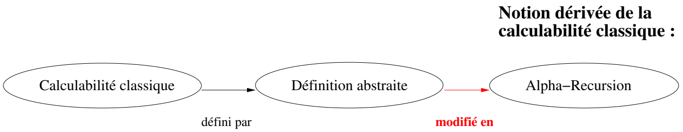
1. *A is computable iff A is Δ_1-comprehensible in HF,*
2. *A is recursively enumerable iff A is Σ_1-comprehensible in HF,*

1. Can be extended to $A \subseteq \mathrm{HF}$;
2. Can be modified by replacing HF by a well chosen set.

Théorème

Let $A \subseteq \mathrm{HF}$, then :

1. *A is computable iff A is Δ_1-comprehensible in HF,*
2. *A is recursively enumerable iff A is Σ_1-comprehensible in HF,*



What's next

- We have a definition, parametrized by a set,
- to modify it we need to find the sets for which the definition stays interesting ;
- we will use Godel's constructibles.

Introduction to Godel's constructibles

$$\mathbb{N},$$

$$\{n \in \mathbb{N} : n \text{ is even}\},$$

$$\{n \in \mathbb{N} : n \text{ is prime}\},$$

$$\{n \in \mathbb{N} : \text{the } n\text{-th diophantine equation has a solution}\},$$

$$\{n \in \mathbb{N} : \phi(n)\} \text{ where } \phi \text{ is a formula.}$$

Remarks :

1. Are there any other sets than these ?
2. Maybe there are a lot ?

Introduction to Godel's constructibles

$$\mathbb{N},$$

$$\{n \in \mathbb{N} : n \text{ is even}\},$$

$$\{n \in \mathbb{N} : n \text{ is prime}\},$$

$$\{n \in \mathbb{N} : \text{the } n\text{-th diophantine equation has a solution}\},$$

$$\{n \in \mathbb{N} : \phi(n)\} \text{ where } \phi \text{ is a formula.}$$

Remarks :

1. Are there any other sets than these ?
   - Yes, by cardinality... An example ?
2. Maybe there are a lot ?
   - As a study, we can try to have the least possible such sets

A universe of sets with no superfluous : strategy

Idea

1. If we have nothing, we have no superfluous
2. If we have something, M, we need to have the sets shaped like :

$$\{x \in M | \phi(x, p)\}$$

   for every formula ϕ and parameters p in M.

•

A universe of sets with no superfluous : strategy

Idea

1. If we have nothing, we have no superfluous
2. If we have something, M, we need to have the sets shaped like :

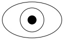
$$\{x \in M | \phi(x, p)\}$$

   for every formula ϕ and parameters p in M.

A universe of sets with no superfluous : strategy

Idea

1. If we have nothing, we have no superfluous
2. If we have something, M, we need to have the sets shaped like :
$$\{x \in M | \phi(x, p)\}$$
for every formula ϕ and parameters p in M.



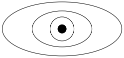
A universe of sets with no superfluous : strategy

Idea

1. If we have nothing, we have no superfluous
2. If we have something, M, we need to have the sets shaped like :

$$\{x \in M | \phi(x, p)\}$$

for every formula ϕ and parameters p in M.



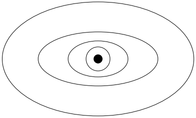
A universe of sets with no superfluous : strategy

Idea

1. If we have nothing, we have no superfluous
2. If we have something, M, we need to have the sets shaped like :

$$\{x \in M | \phi(x, p)\}$$

for every formula ϕ and parameters p in M.



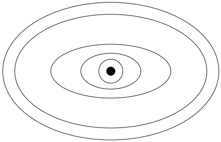
A universe of sets with no superfluous : strategy

Idea

1. If we have nothing, we have no superfluous
2. If we have something, M, we need to have the sets shaped like :

$$\{x \in M | \phi(x, p)\}$$

for every formula ϕ and parameters p in M.



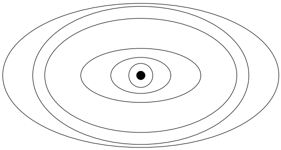
A universe of sets with no superfluous : strategy

Idea

1. If we have nothing, we have no superfluous
2. If we have something, M, we need to have the sets shaped like :

$$\{x \in M | \phi(x, p)\}$$

for every formula ϕ and parameters p in M.



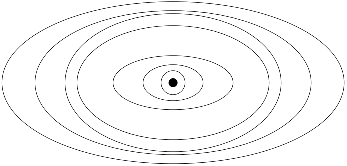
A universe of sets with no superfluous : strategy

Idea

1. If we have nothing, we have no superfluous
2. If we have something, M, we need to have the sets shaped like :

$$\{x \in M | \phi(x, p)\}$$

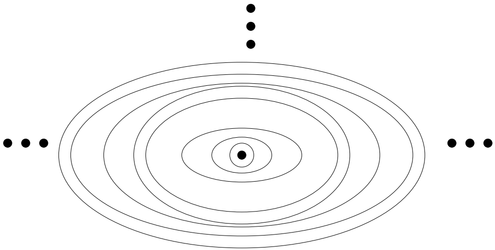
for every formula ϕ and parameters p in M.



A universe of sets with no superfluous : strategy

Idea

1. If we have nothing, we have no superfluous
2. If we have something, M, we need to have the sets shaped like :
$$\{x \in M | \phi(x, p)\}$$
for every formula ϕ and parameters p in M.

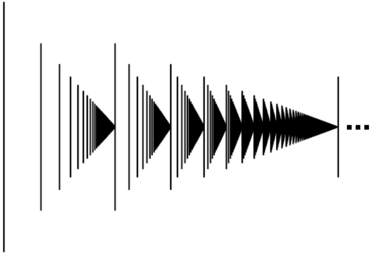


Ordinals

Définition

An ordinal is a set α such that

1. α is transitive : $\forall x \in \alpha, \forall y \in x, y \in \alpha$
2. $(\alpha, \in)$ is a well ordering.



- Some ordinals are successors,
- some ordinals are limits.

A precise definition

Gödel's constructible universe (1938)

Gödel's constructible at rank α, written L_α are defined by induction alons ordinals :

1. $L_0 = \emptyset$,
2. $L_{\alpha+1} = \mathrm{Def}(L_\alpha)$,
3. $L_\lambda = \bigcup_{\alpha<\lambda} L_\alpha$.

The constructibles are the elements of $\bigcup_\alpha L_\alpha$.

Définition

$$Def(M) = \left\{ E^M_{\phi,\bar{p}} : \phi \text{ is a formula and } \bar{p} \in M \right\}$$

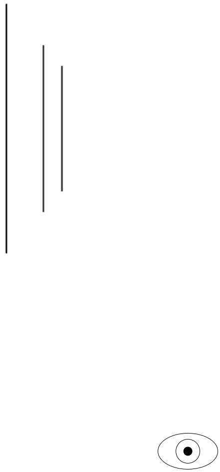
where

$$E^M_{\phi,\bar{p}} = \{x \in M : \phi(x, \bar{p}) \text{ is true in } M\}$$

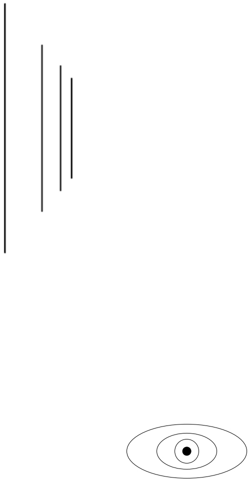
Illustration

Illustration

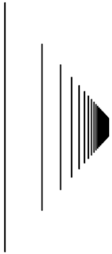
Illustration

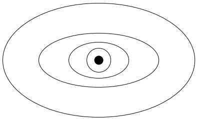


Illustration

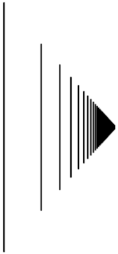


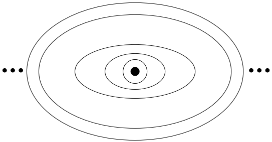
Illustration



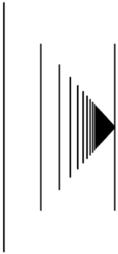


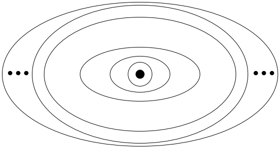
Illustration



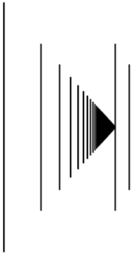


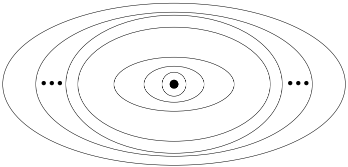
Illustration



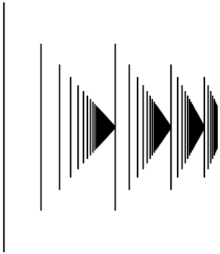


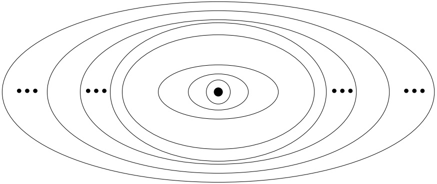
Illustration



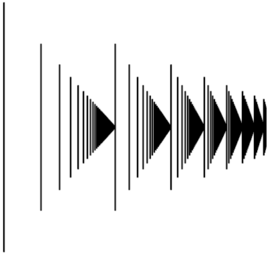


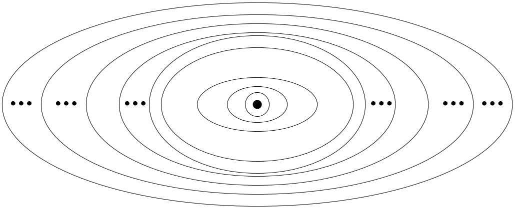
Illustration



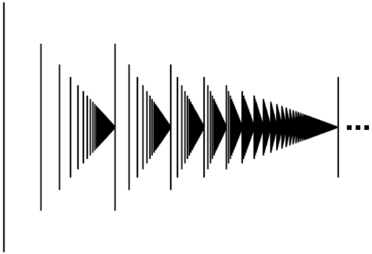


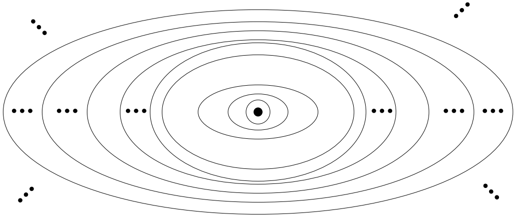
Illustration





Illustration





Examples

The constructibles are constructed layer by layer. These are some particular layers :

1. $L_{n+1} = \mathcal{P}(L_n)$ for n an integer ;
2. $L_\omega = \mathrm{HF}$, the hereditarily finite sets ;
3. $L_{\omega_1^{CK}} = \mathrm{HYP}$, the sets with hyperarithmetic codes ;
4. $L_\lambda = \mathrm{WRT}$, the sets with writable codes.

Examples

The constructibles are constructed layer by layer. These are some particular layers :

1. $L_{n+1} = \mathcal{P}(L_n)$ for n an integer ;
2. $L_\omega = \mathrm{HF}$, the hereditarily finite sets ;
3. $L_{\omega_1^{CK}} = \mathrm{HYP}$, the sets with hyperarithmetic codes ;
4. $L_\lambda = \mathrm{WRT}$, the sets with writable codes.

We find again HF !

Examples

The constructibles are constructed layer by layer. These are some particular layers :

1. $L_{n+1} = \mathcal{P}(L_n)$ for n an integer ;
2. $L_\omega = \mathrm{HF}$, the hereditarily finite sets ;
3. $L_{\omega_1^{CK}} = \mathrm{HYP}$, the sets with hyperarithmetic codes ;
4. $L_\lambda = \mathrm{WRT}$, the sets with writable codes.

We find again HF !

Théorème

Let $A \subseteq \mathbb{N}$, then :

1. *A is computable iff A is Δ_1-comprehensible in L_ω,*
2. *A is recursively enumerable iff A is Σ_1-comprehensible in L_ω,*

Computability in a space of sets

The basic definition of α-recursion :

Definition

Let α be an ordinal and $A \subseteq L_\alpha$. We say that :

1. A is α-finite if $A \in L_\alpha$;
2. A is α-recursive if A is Δ_1-comprehensible in L_α ;
3. A is α-recursively enumerable if A is Σ_1-comprehensible in L_α.

- Some α will reveal more intersting than others,
- A is a set of α-finite elements, not only integers.

Intuition

We see a computation as a search into all the α-finite sets.

Admissibility I

It is not yet finished ! Because :

Remark

Some α will reveal more intersting than others...

- Which α ?
- Then, what are the properties of L_α ?

Admissibility I

It is not yet finished ! Because :

Remark

Some α will reveal more intersting than others...

- Which α ?
  - The admissibles ordinals, the ω_1^X for any $X \in 2^\omega$.
- Then, what are the properties of L_α ?
  - L_α is then admissible, it verifies the Kripke Platek axioms : L_α is a model of Δ_1-comprehension et Σ_1-collection.

Admissibility II

Définition

- A set is said admissible if it verifies the Kripke-Platek axioms, of which the most notable are Δ_1-comprehension ans Σ_1-collection.
- An ordinal α is said to be admissible if L_α is admissible.

- L_ω, $L_{\omega_1^{CK}}$, L_λ are admissibles.
- If α is admissible, the mapping of an α-finite by a function of α-recursive graph is α-finite.

Intuition

An ordinal α is admissible if the α-recursion is not too far from computability.

What did we defined ?

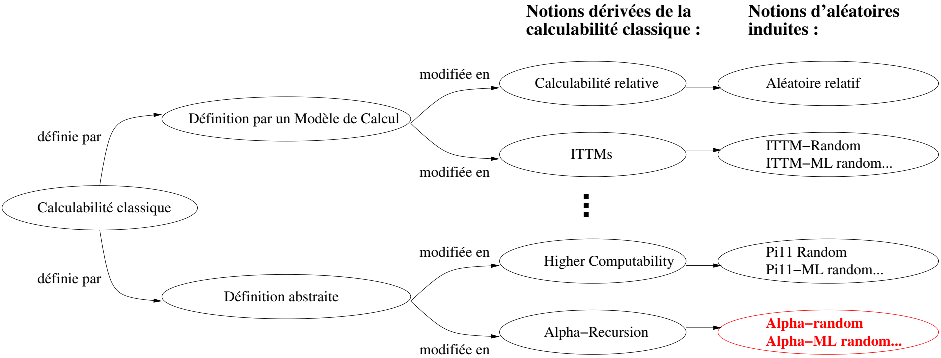
Intuition

We see a computation as a search into all the α-finite sets.

- ω-recursion, is classical computability ;
- ω_1^{CK}-recursion, is higher computability ;
- λ-recursion, is ITTM computability.

We have a general and satisfying definition of computability.

Randomness Part



Defining randomness...

A randomly chosen sequence of bits

0 1 1 1 0 0 1 1 1 1 1 0 1 0 0 0 1 0 1 0 1 . . .

There exists several paradigms to define what it is to be random for a sequence of bits :

1. Impredictability,
2. Incompressibility of prefixes,
3. No exceptionnal properties.

We will use the third paradigm.

Algorithmic randomness ?

Question

For $X \in 2^\omega$, what does it means for X to be a random set ?

Algorithmic randomness ?

Question

For $X \in 2^\omega$, what does it means for X to be a random set ?

1. Has no more even numbers than odd ones,
2. is not computable,
3. Is not like $b_0 0 b_1 0 b_2 0 \ldots$

We define randomness by the negative : we remove those which do not seem random.

Formally

Paradigme

X is random if X has no exceptionnal property

Becomes

Définition

X is $\mathscr{C}$-random if $\forall P \in \mathscr{C}$ such that $\lambda(P) = 0$, $X \notin P$

Formally

Paradigme

X is random if X has no exceptionnal property

Becomes

Définition

X is $\mathscr{C}$-random if $\forall P \in \mathscr{C}$ such that $\lambda(P) = 0$, $X \notin P$

Examples of $\mathscr{C}$:

1. the null Π^0_2,
2. the null Δ^1_1,
3. the Martin-Löf tests...

$\mathscr{C}$ countable ensures us that the $\mathscr{C}$-randoms are conull.

Martin-Löf Random

- Martin-Löf randomness has been the most studied.
- It has a definition for every of the three paradigm : impredictability, incompressibility of prefixes, and no exceptionnal properties.

Martin-Löf Random

- Martin-Löf randomness has been the most studied.
- It has a definition for every of the three paradigm : impredictability, incompressibility of prefixes, and no exceptionnal properties.

Définition (Martin-Löf's tests)

A Martin-Löf test is an intersection $\bigcap_n \mathcal{U}_n$, where $(\mathcal{U}_n)$ is recursively enumerable, and $\lambda(\mathcal{U}_n) \leq 2^{-n}$.

Also called Π^0_2 effectively null.

Définition (Martin-Löf Random)

X is Martin-Löf Random if X do not belong to any Martin-Löf test.

α-randomness

Following this principle, we define the tests in L_α.

Définition

X is random over L_α (or α-random) if X do not belong to any null borel set with code in L_α.

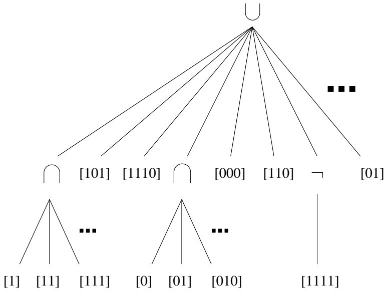


Figure – A borel code

- ω_1^{CK}-randomness is Δ^1_1-randomness,
- λ-randomness is ITTM-randomness.

α-ML-randomness

We continue the process to generalise Martin-Löf's idea :

Définition

- An α-ML test is a Martin-Löf test $\mathcal{U} \subseteq \omega \times 2^{<\omega}$ which is α-recursively enumerable.
- X is α-ML random if it is in no α-ML tests.

- ω-ML randomness is ML random,
- ω_1^{CK}-ML randomness is Π^1_1-ML randomness,
- λ-ML randomness is $\mathrm{ITTM}_{\mathrm{ML}}$ randomness

A question

Question

For every α, do the notions of "α-random" and "α-ML random" coincide ?

A question

Question

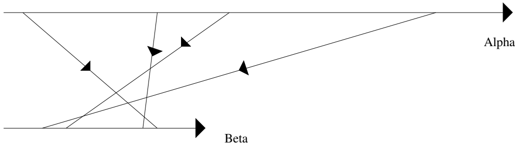
For every α, do the notions of "α-random" and "α-ML random" coincide ?

Théorème

Δ^1_1-randomness and Π^1_1-ML randomness are different notion.

This answers the quesion in a particular case. We would like a condition on α for it to be true.

Projectibility



Définition

α is projectible into β if there exists anα-recursive function, one-one from α to β.

- ω_1^{CK}, λ are projectible into ω ;
- not every ordinals are projective into a smaller ordinal thant themselves.

An equivalence

Théorème

The following are equivalent :

1. α *is projectible into* ω, *and*
2. α*-randomness and* α*-ML randomness are different notions.*

Being projectible into ω allows us to reduce "space" and "time" into a single dimension.

Corollaire

ITTM-randomness et ITTM-ML randomness are two different notions.

Conclusion

- L'α-recursion extends computability, and includes other extensions ;
- it allows us to define new notions of randomness ;
- we have an equivalence between a property of set theory and a property of algorithmic randomness.

Thanks for your attention !