Universality everywhere and beyond, an epic of computer science

Maurice Margenstern

Université de Lorraine
LITA, EA 3097

Turing in Context II

October, 12, 2012

Royal Flemish Academy of Belgium
for Science and the Arts

in this talk:

Prologue

1. Universality and Church thesis
2. Turing-complete models
3. Beyond the halting barrier
4. Computer science: a revolution

Prologue

Church thesis is something about
a barrier
man like barriers he puts
everywhere and first
onto himself
now, when man sees a barrier his persistent goal is to destroy it

Prologue

interestingly
when man cannot immediately annihilate the barrier, he **dreams** about that, about the way to use for that goal, about what might happen once the barrier is destroyed

Prologue

when eventually man removes
the barrier, how he did does
not coincide with his first dreams
and what happens is always a
surprise far beyond his darest
dreams

Prologue

we have an interesting phenomenon
about this:
man cannot fly like birds
still, man has thousand years
dreamed to fly
now, man flies, but not like
birds

Prologue

the way man found is very
special
presently, man could probably
imitate the fly of birds but he
looks at this possibility rather
reluctenlty
he is no more that interested

Prologue

this metaphore about the fly of birds applies to many similar situations
I think that it applies too to the barrier implied by Church thesis
it also applies to artificial intelligence

Prologue

we shall try to justify this opinion
first we look at several aspects of Church thesis and **then** we shall look at the different dreams to overcome it as, presently, we are still in the time of dreams
then we look at the contribution of computer science to philosophical questions

1. Church thesis

1.a The thesis itself

1.b Universal Turing machines

1.c Algorithms

1.d bottom-up versus top-down

1.e questions

on the Turing machine

1.a The thesis itself

as already known, the thesis says:

any intuitive algorithm can be translated into a formal algorithm with the same outputs, given the same inputs

1.a The thesis itself

important remark,

the thesis cannot be a theorem

as it speaks about an *intuitive* concept

so, it cannot be proved

1.b Universal Turing machines

a Turing machine:

tape and head:

finite alphabet A, finite states, $1..N$

program: finite sequence of **instructions**

format of an instruction: $\mathbf{q}xy\mathtt{M}\mathbf{p}$

q current state, x read
p new state, y written,
M new position of the head

1.b Universal Turing machines

configuration at t:

C_t describes the tape, the state and the position of c

from C_t to C_{t+1}:

apply at c the unique possible instruction if any, otherwise stop

initial configuration:C_0

halting: no instruction or specific one

the machine may not halt

1.b Universal Turing machines

working of **U** simulating **M**:

encode **M** program as P:

state by state,
instruction by instruction:

$\texttt{XY}cod_{1,1}\ldots\texttt{Y}cod_{1,k}\texttt{X}\ldots\texttt{XY}cod_{n,1}\ldots\texttt{Y}cod_{n,k}$

encode **M** data as D:

$\texttt{U}cod(a_1)\ldots\texttt{U}cod(a_s)$

1.b Universal Turing machines

U must mimic **any** **M**

encode **M** alphabet and states:

i^{th} letter: $\mathtt{1}^i$; j^{th} state: $\mathtt{a}^j$

$cod = \mathtt{1}^{|y|}\mathtt{Ma}^{|\mathbf{p}|}$, $\mathtt{M} \in \{\mathtt{L},\mathtt{R},\mathtt{S}\}$

$|y|$: rank of y in **M** alphabet

$|\mathbf{p}|$: rank of **p** in **M** states

1.b Universal Turing machines

2 markers $\mathtt{W}$:

$\mathtt{W}_P$ on current instruction

$\mathtt{W}_D$ on current square

head of **U**:

loop of
motion from $\mathtt{W}_P$ to $\mathtt{W}_D$ and back

1.b Universal Turing machines

important corollaries:

the halting barrier:

undecidable problems
unsolvable problems

no Turing machine can say from the code of **M** and its data D whether **M** halts its computation from D

1.b Universal Turing machines

stronger result:

let **U** be a universal Turing machine

its halting on its data is undecidable

1.c **Algorithms**

a possible formalism:

three **constructors**:

juxtaposition

branching

loop

juxtaposition:

if A and B are algorithms,

so is $A; B$;

which means:

first execute A,
when completed, execute B

branching:

if A, B and C
are algorithms with C
a two-valued one
then:

if C
then $\{A\}$;
else $\{B\}$;

is an algorithm

branching:

which means:

compute C
if C yields 1st value
execute A
otherwise, execute B

loop:

if A and C
are algorithms with C
a two-valued one
then:

while C
$\{A\}$;

is an algorithm

loop:

which means:

compute C
if C yields 1st value
execute A and then,
repeat
otherwise,
if C yields 2nd value
execution completed

basic point

finitely many

elementary actions

at this moment two possible interpretations:

bottom-up approach

top-down approach

1.c bottom-up versus top-down

bottom-up approach,

the standard approach

given:
- finitely many integer valued variables
- operations:
  - affectation,
  - addition, multiplication
  - subtraction, division
  - test of zero

bottom-up approach,

in fact, incrementing by 1

is enough to get universality:

There is a formal algorithm able to simulate any formal algorithm

the other operations give
no more power, just
more comfort and efficiency

bottom-up approach,

from the universality theorem:

again undecidability results
the same as
for Turing machines

top-down approach,

the idea is that the notion
of **elementary action**
may be considered as **relative**

let us look at an example

top-down approach,

an example from **do it yourself**

it is desired to nail a thin board T
with ends A and B on a thick
one P

top-down approach,

algorithm:

```
nail A on P; nail B on P;
go to A;
while not finished
{advance to B by 20cm;
put a nail at this point;};
```

here, **finished** and **put a nail**

are **elementary actions**

top-down approach,

we can develop the action of nailing

algorithm:

```
place the nail on the spot;
while not finished
{ give a hit with the hammer;};
```

here, **give a hit** is elementary

top-down approach,

now, a hit with the hammer can also be divided into elementary actions:

imagine that you have to make a cartoon about the perfect do it himself where several lectures are devoted to nailing a board upon another one

top-down approach,

division up to which point?

an interesting question:

describing the motion of the hammer from the initial position of the hand until it reaches the head of the nail leads us straightforward to Zeno paradoxes

first intermezzo,

Zeno paradoxes

one of the paradoxes:

Achileus and the turtoise

it says that Achileus never reaches the turtoise who stands at a given distance from him

Achileus has before to reach the mid-point of this interval but before, again the mid-point of this new goal and so on

Zeno paradoxes

Zeno's conclusion: Achileus never starts from his position

we have to be careful: we know the paradoxes not from Zeno directly

we do not know his initial position and we do not know whether he commented his conclusion

Zeno paradoxes

traditional explanation:

18^{th} maths solved the problem by the convergence of series, here $\sum_{k=1}^{\infty} 2^{-k} = 1$

this argument does not answers the question

only $\lim_{k \to \infty} \frac{1}{2^k} = 0$ is in agreement with Zeno's statement

Zeno paradoxes

correct explanation:

assumption of Zeno's statement: time and space are both continuous

as conclusion contradicted by experience, the assumption is wrong

as time is connected to space by motion, time and space must be both discrete

Democrites developped the first atomistic theory

bottom-up versus top-down

conclusion of the discussion:

in a top-down approach,
the refinment cannot be
continued endlessly

bottom-up versus top-down

this grounds the interest of the bottom-up approach

in particular the elementary actions as defined above

bottom-up versus top-down

however,
an interesting question:

in top-down approach, is the
refinment uniformly bounded?

1.e. questions on TM's

initial configuration:

not arbitrary:
it might be a universal set

basic standards:

finite or at most primitive recursive (Minsky)

when not finite, at most context free

1.e. questions on TM's

initialization

and now:
how is the tape initialized?

from definition, it cannot be by another Turing machine,

otherwise, infinite sequence of Turing machines initializing the next one

$\Rightarrow$ initialization, a process outside the theory

1.e. questions on TM's initialization

a weak point of Church thesis

the physical Super Church thesis, namely the world is a big Turing machine, has the same defect:

who initialized or how it was initialized?

a hint to the answer is in the question

1.e. questions on TM's

yardstick for museum only?

Turing machine:

first device shown to be universal

used as yardstick in complexity theory

however,
considered by many people

as a museum object

a riddle:

$TGTCCACACACACAACTCCACACAACTCCACAAGTCCACAACCTCTTCCAAG$
$TGTCCACAAGCTCTCTCTTCACTCCACAACTCACTCAC$
$TGTCACCTCTCTCTCTTCCACACACAAGCTCTCTCTTCCACAACTCACTCAC$
$TGTCCACACACACAAGCTCTCTTCCACACAAGTCCACAACCTCTTCAGTCAG$
$TGTCTCTCACTCCACAACCTTCCAAC$

what is this?

a riddle

the answer: an encoding of a small universal Turing machine

	b	0	1	c	d
1	dR	$1R$	$0L$	$0R2$	bL
2	$0L4$	R	$0R$	R	R
3	$R5$	$cL4$	$0R$	R	R
4	$dL3$	$1L$	$0R2$	L	L
5			R	$1R1$	bR

Yu. Rogozhin, TCS **168**(2), (1996), 215-240

a riddle

indeed

encoding of the letters:

b	0	1	c	d
CA	CACA	CACACA	CACACACA	CACACACACA

states: 1 up to 5:

CT CTCT CTCTCT CTCTCTCT CTCTCTCTCT

directions: L R

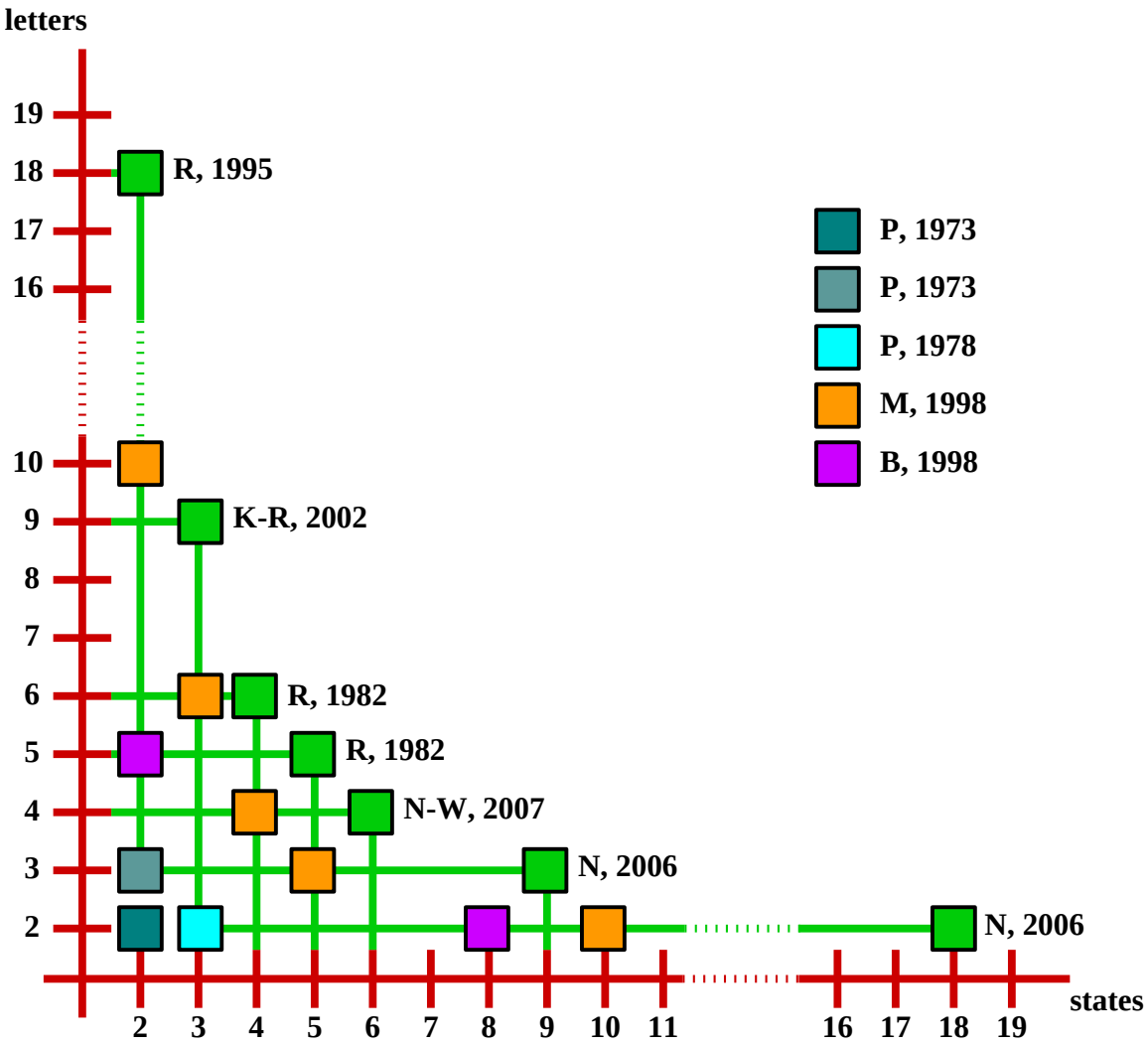
AG AC

markers: instruction states

TC TG

parenthesis

small universal Turing machines



1.e. questions on TM's

yardstick for museum only?

this looks like simpler than the smallest virus

consequently, simpler than a bacteria

hence, than a colony of bacteria

second intermezzo,

colonies of bacteria

what Professor Ben Jacob says

Eons before humans, bacteria inhabited a very different Earth. As the earliest life form they devised ways to counter the spontaneous course of increasing entropy and convert high-entropy, inorganic substances into low entropy, organic molecules...

E. Ben Jacob, Social behavior of bacteria...
European Physical Journal B, **65**(3), (2008), 315-322

what Professor Ben Jacob says

To change environmental hazards, bacteria resort to a wide range of cooperative strategies...

They collectively glean information from the environment, communicate, distribute tasks, perform distributed information processing and learn from past experience.

E. Ben Jacob, Social behavior of bacteria...
European Physical Journal B, **65**(3), (2008), 315-322

it is known that bacteria
can be found everywhere
in almost every possible hard
conditions as

geysers,
ocean fathoms,
core of the earth

and even

atomic piles

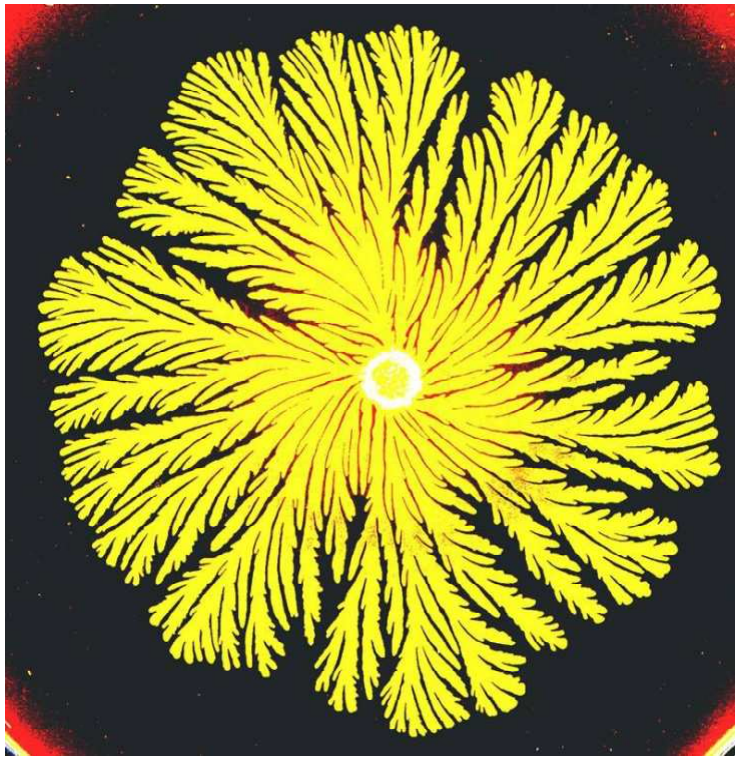
the experiments of Professor Ben Jacob

the idea was to put

colonies of bacteria

into **severe** conditions

the experiments of Professor Ben Jacob



images by courtesy of Pr. Ben Jacob

The Turing machine: yardstick for museum only?

the smallest Turing machines
invites us to reconsider
our look at bacteria
and to consider the arithmetic hierarchy
as the today best tool
to classify colonies of bacteria

2. Turing-complete models

2. Turing-complete models

we sall see:

- 2.a **the classical models**
- 2.b **the formal models**
- 2.c **the circuit models**
- 2.d **the bio-inspired models**
- 2.f **the railways model**

2.a the classical models

just to remember:

recursive functions
λ-calculus

fundamental convergence result of 1936:

each of these models is equivalent to Turing machines

all next models are also equivalent to Turing machines

most often people ‘prove’ the equivalence
by Church thesis

this is a very bad attitude

laziness is not an excuse

2.b the formal models

first, algebraic-like models:

Post systems, 40's years

later, 60's and 70's:

second, linguistic models:

grammars,

in connection with Chomsky's hierarchy

2.c the circuit models

these models implement

the register machine, 60's

very popular model

most universality results are implementations of a register machine

register machines

'close' to assembly language

but essentially different:

finitely many registers
finite list of instructions,

called the **program**

register machines

registers, R_1, ..., R_k:

each one contains an integer,

not a machine integer

3 formats for an instruction:

lab **inc** R_k
lab_0 **dec** R_k, lab_1
lab_0 **jump** lab_1

register machines

working:

lab **inc** R_k:

$R_k := R_k + 1;$

lab_0 **dec** R_k, lab_1

if $R_k > 0$ then $R_k := R_k - 1;$

else goto $lab_1;$

halting: lab_0 **jum** $lab_{last} + 1;$

circuit models

two parts:

the registers, usually two of them

the sequencer:

it implements the program

circuit models

the sequencer is implemented

by using logical gates as

in electronic circuits,

mainly AND, OR and NOT gates

alghough NAND gates are enough

circuit models

a register is implemented as an infinite sequence of units dispatched along a line of the Euclidean plane

each one contains one bit and can be accessed either for reading or for writing

at initial time all units but finitely many ones are set to 0

2.d the bio-inspired models

several very different models:

neurons,
McCullock and Pitt, 1941

cellular automata,
von Neumann, Ulam, 1949

DNA computing,
Head, 1982

membrane computing,
Paŭn, 1999

cellular automata

inspired by the organization of tissues into cells in most beings

first goal of von Neumann, to obtain both:

- universality of computation
- self-reproduction

he obtained such a CA with 29 states

cellular automata

dropping universality, very simple self-reproducing CA's can be obtained

famous example:
Langton's loop

cellular automata

cellular automata must fulfill
three requirements:

uniformity

of distribution in space
of neighbourhood around each cell
of computing ability in each cell:
same finite automaton

cellular automata

the automaton in each cell:

input:
state of the cell + neighbours' states

output: the new state

discrete clock
at each tick of the clock
uniform updating of the states

cellular automata

in the Euclidean plane,
usually the square grid with
two kinds of neighbourhood:

together with the considered
cell,

in von Neumann neighbourhood:
cells N, S, E, W

in Moore neighbourhood
also cells NE, NW, SE, SW

cellular automata

important results about universality:

2 states with Moore neighbourhood
Rendell, 2011

2 states but weak universality
as initial infinite configuration,
Conway, game of life, 1982

cellular automata

also on the line:

neighbours: L, R

strong universality: 7 states,
Nordhal-Lindgren 1991

weak universality: 2 states,
M. Cook, 2004

cellular automata

also:

extraordinary power of simulation

CA's used to simulate:
diffusion-reaction phenomena
percolation phenomena
wearing of brake pads in cars
traffic of cars on highways
circulation of peatons in railway stations
colonies of bacteria
life of the skin

cellular automata

coding and simulation are the two pillars of computer science, both rooted in discrete computations

coding should be considered as a primitive term

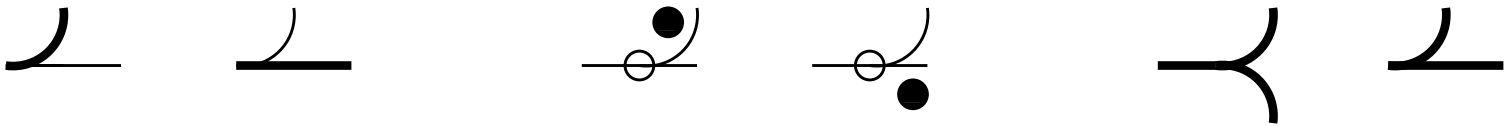
simulation too, at least in first approximation

the railway model

circuit with a single **locomotive**

tracks and switches

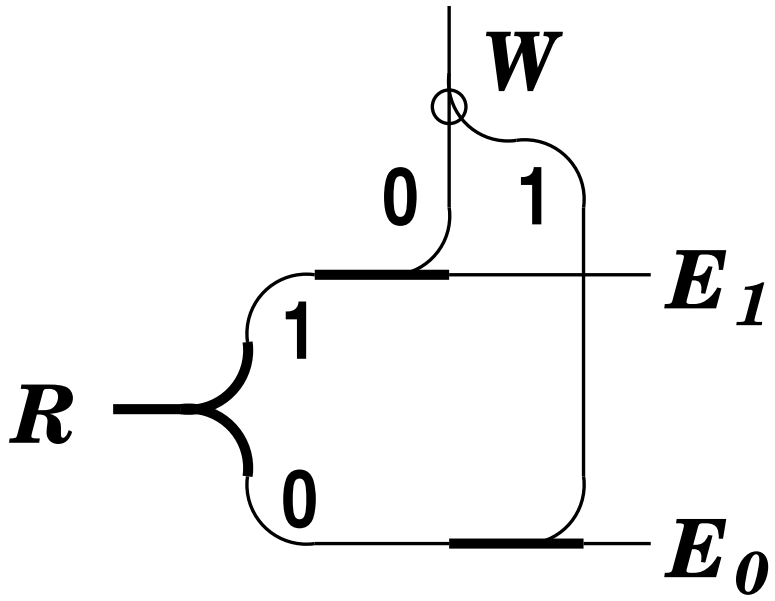
the switches:



the railway model

the basic element,

1 bit of information:



3. Beyond the halting barrier

3. Beyond the halting barrier

we sall see:

3.a the pionnier: Brower

3.b Turing and Kleene too

3.c real numbers

3.d time

3.e space

3.a Brower and the intutionism

intuitionism: born in the 20's

strong requirements of effectivity

however, the 'fan theorem'

strongly non-effective result

Brower: the creative subject

the creative subject models the creation of mathematics by mathematicians

it is truely constructive and it is not a formalized algorithm, so that this contradicts Church's thesis

Brower: the creative subject

remark:

Brower's theory explicitly uses a discrete time and there is no initialization problem...

3.b Turing and Kleene too

Turing:
machines with an **oracle**

at some point, the machine is allowed to ask a question whose answer is supplied by an oracle

formally, there is another read only tape containing an appropriate set of values which the machine can freely read

Turing and Kleene too

Kleene:
relative recursivity:

besides the traditional basic functions another kind of basic functions is accepted, say f

the new functions are called partial recursive in f

Turing and Kleene too

machines with oracle and relative recursivity allow us to construct the **arithmetical hierarchy**

the halting problem is the second floor of the hierarchy, recursive functions being the first floor

the hierarchy has infinitely many floors

3.c real numbers

it consists in introducing
mathematical real numbers
in the setting of an
algorithmic context

example: the BSS model

looks like a register machine
but:

- registers contain mathematical real numbers
- the equality between real numbers is an elementary operation

the BSS model

problem of this theory:
too powerful

as **oracle**-TM's:

just choose an appropriate oracle

another way:

playing on the initial configuration:

TM's or CA's with arbitrary initial configurations

then possible to encode the characteristic function of the halting problem

but again: initialization, how?

3.d time

time already present in the **creative subject**

several approaches:

- stabilization at the limit
- TM's computing in infinite time

stabilization at the limit

imagine a TM with two tapes:

a working tape
for temporary computations

the result tape:
O or 1 appended as a result
of a partial computations

stabilization at the limit

by definition,

such a TM gives a result if and only if after a certain time, the TM always appends the same symbol on the result tape

result:

easy to see that this **solves the halting problem**

TM's computing in infinite time

again meeting Zeno

the use of ordinals

TM's computing in infinite time

again Zeno:

from pseudo Gordon Moore's prediction:

computers run twice faster every year

TM's computing in infinite time

Moore's 'law' transformed into:
Copeland, 1998

first step: $t_1 = 1$ unit

step n+1: $t_{n+1} = \frac{t_n}{2}$

conclusion:

we go through an infinite computation in finite time

TM's computing in infinite time

from Hogarth and Malamant:

black whole computations:

connected with relativity theory
and acceleration of time around
black wholes

TM's computing in infinite time

the use of ordinals, J.D. Hamkins, 2002

the idea is to expend Zeno computations in infinite time

the use of ordinals

formally:

working as for classical Turing machines from time α to time $\alpha+1$

at a limit ordinal:
where stabilization: limit value
where infinite alternations: 1

the use of ordinals

clearly, the halting problem is decidable in ω-time

extremely powerful theory

a very interesting hierarchy with specific properties

3.*e* **space**

more recently, researches involving **space**

two directions:

a new kind of Zeno computation

the use of another geometry

a new Zeno computation

a new black whole approach, Durand-Lose, 2004:

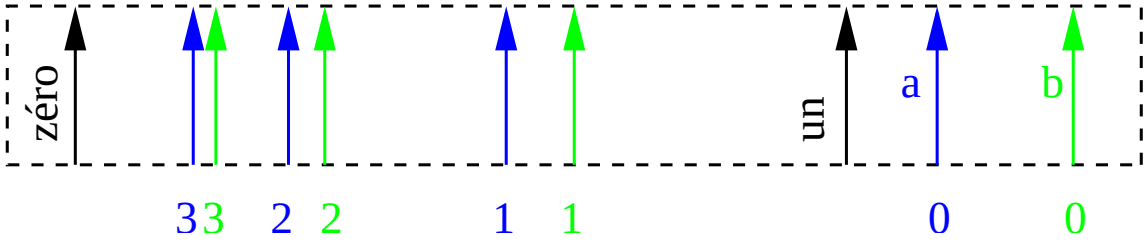
computation by discrete CA-like signals in the Euclidean plane

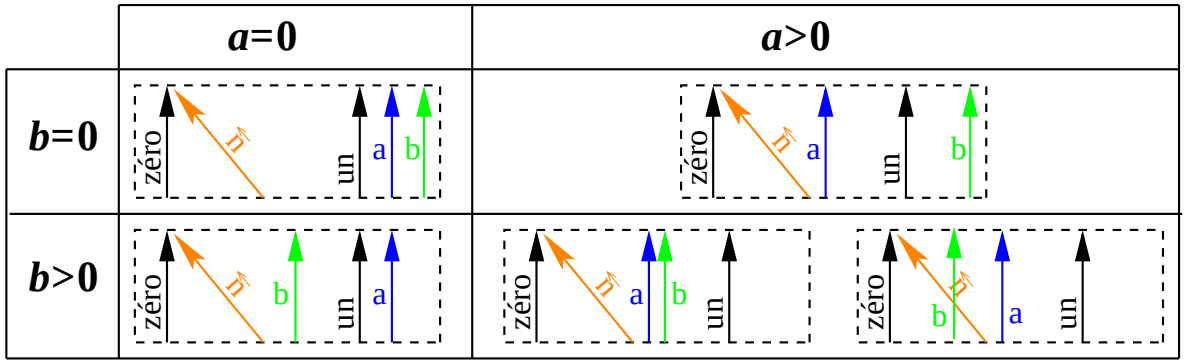
spatial version of 'Moore's law':

$n+1$-step signal: half sizes of n-step

a new Zeno computation

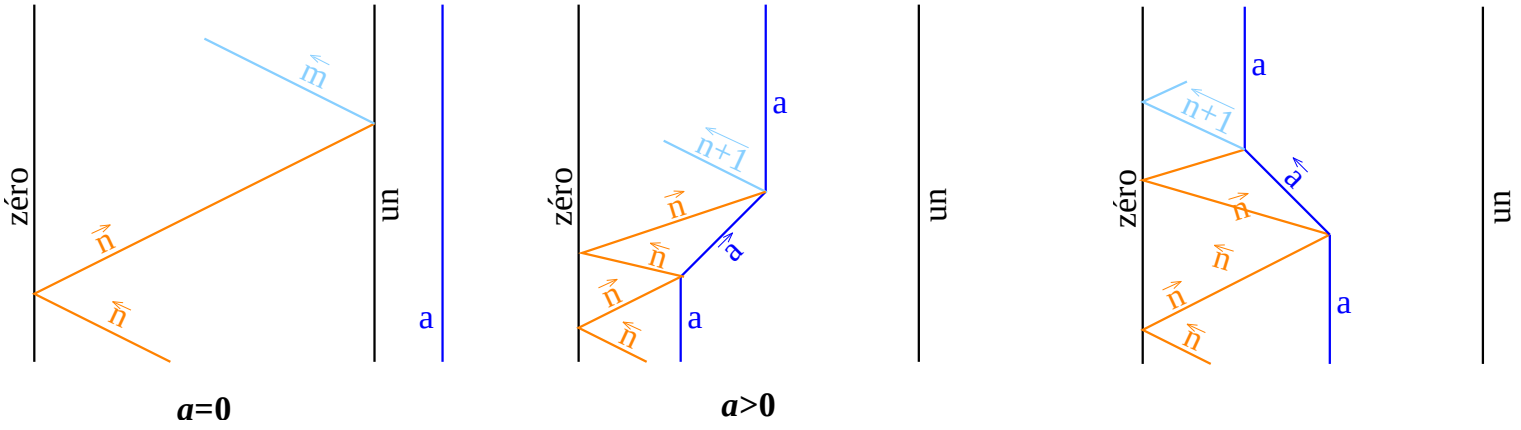
implementation of registers:





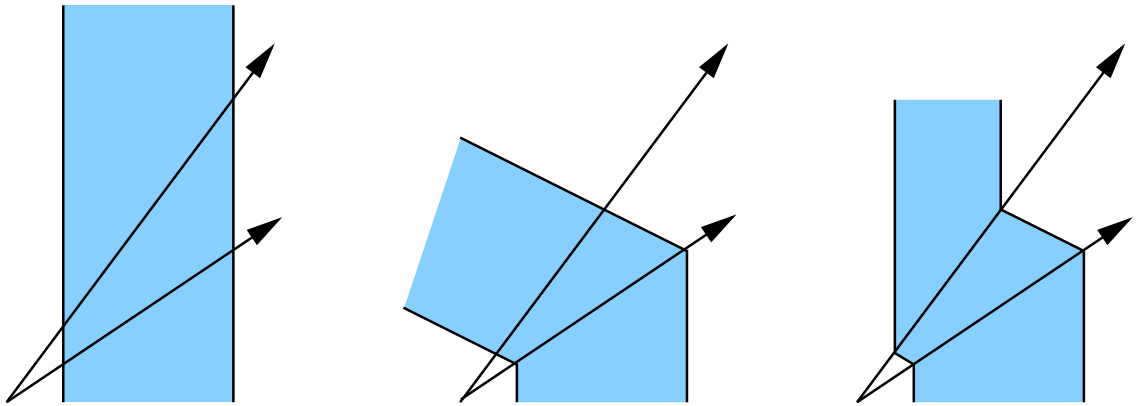
a new Zeno computation

operations on registers:



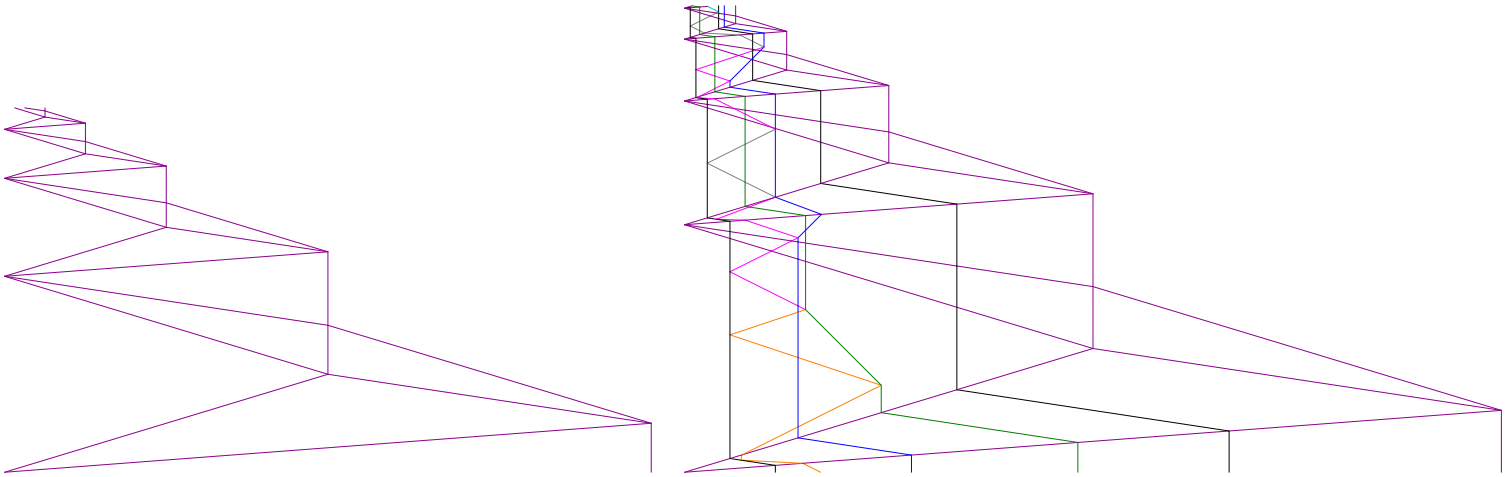
a new Zeno computation

deviation and compression of the signals



a new Zeno computation

deviation and compression of the signals

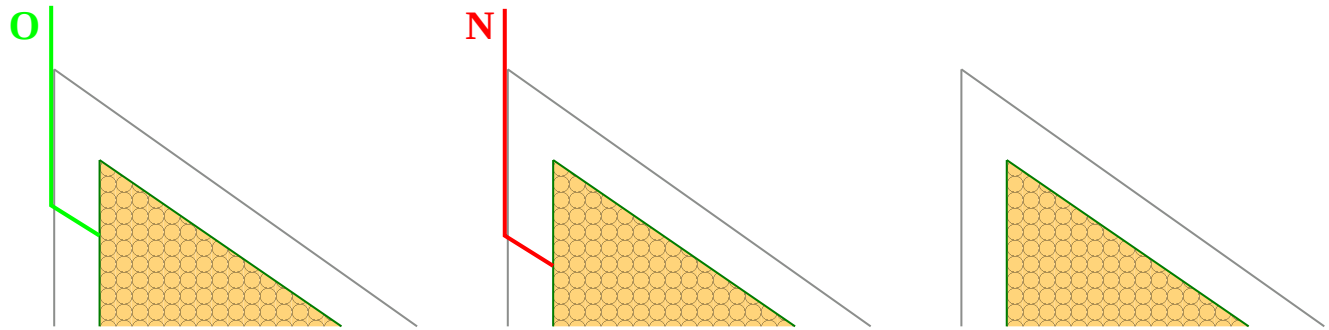


the frame

the result

a new Zeno computation

the black whole effect:



clearly, this solves the halting problem

a new Zeno computation

interest of the model:
it involves rational numbers only

problem:
it requires divisibility at infinity

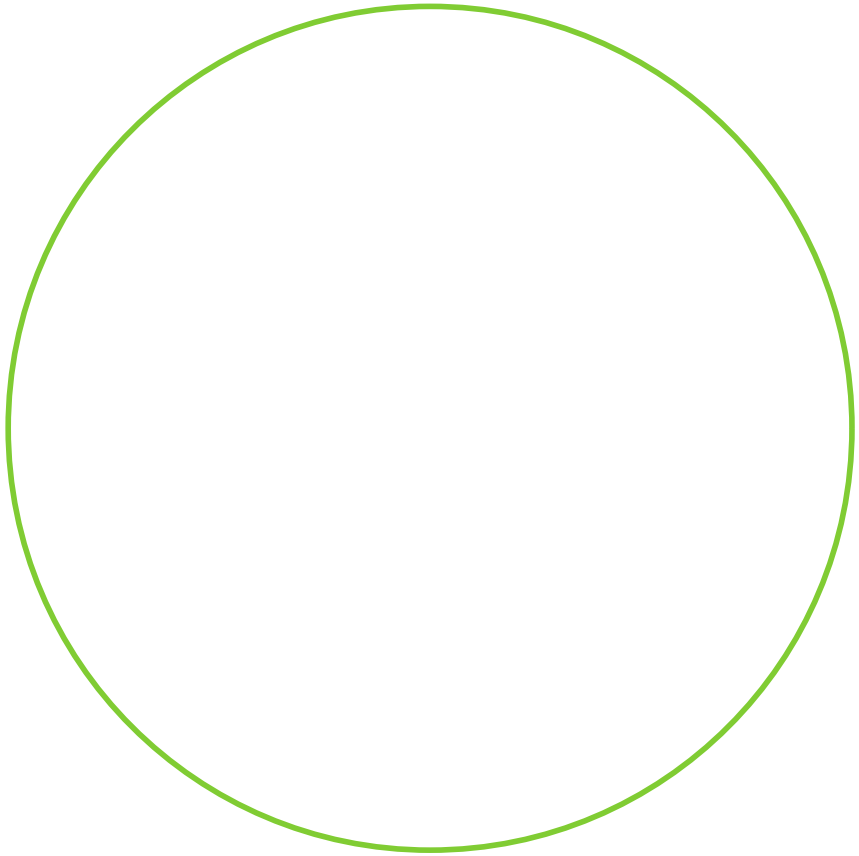
the use of another geometry

third intermezzo,

hyperbolic geometry

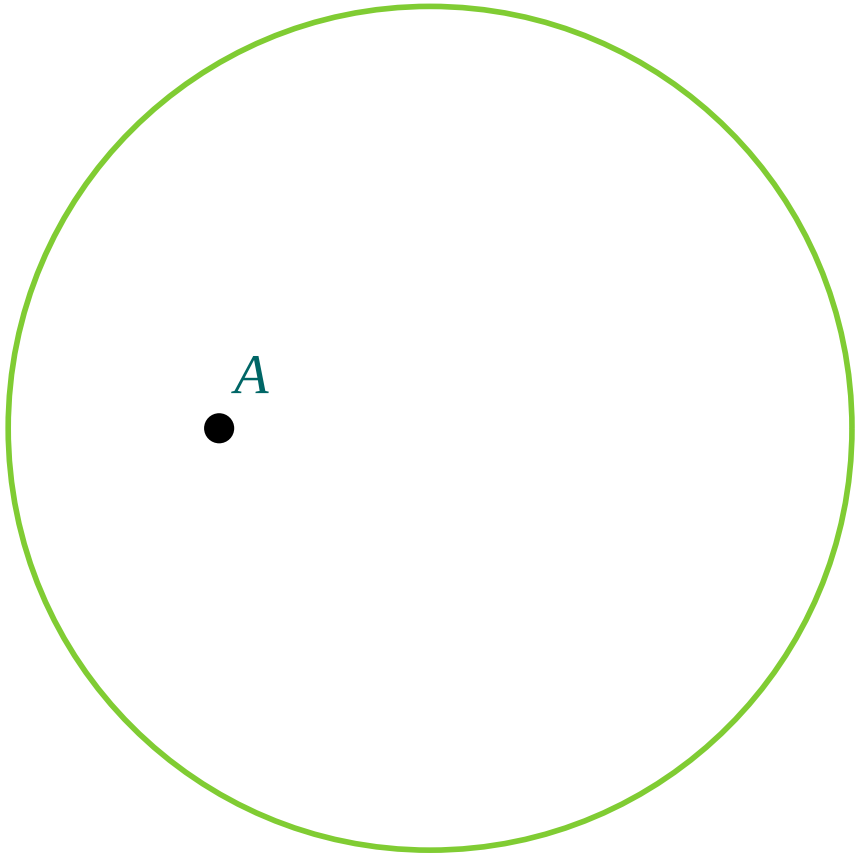
Poincaré's disc model

the disc



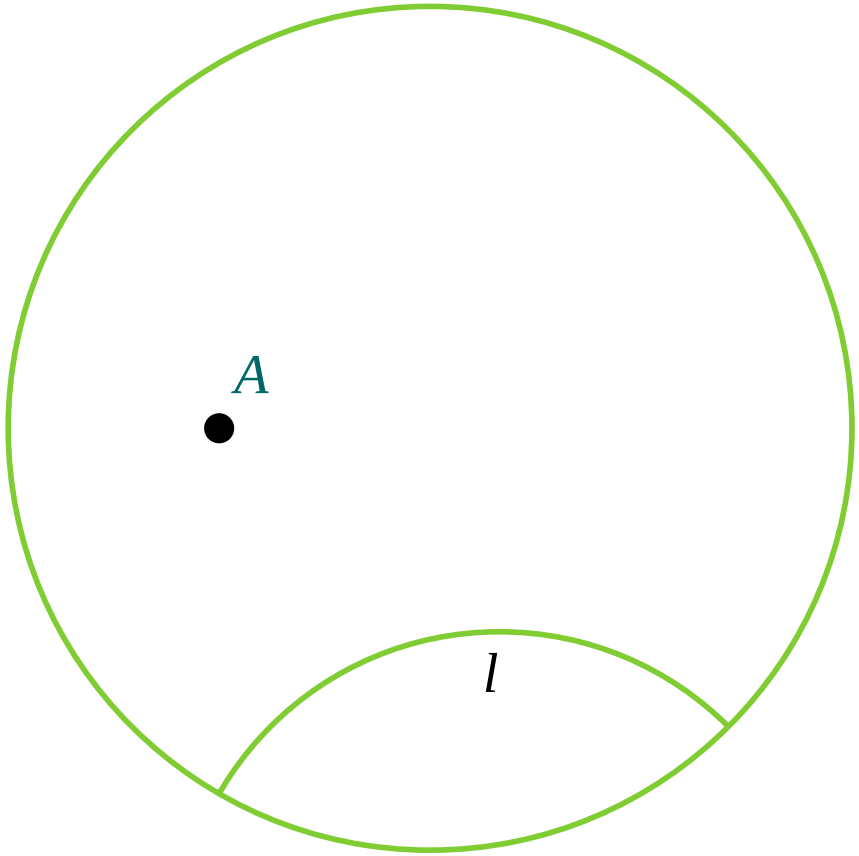
Poincaré's disc model

a point A



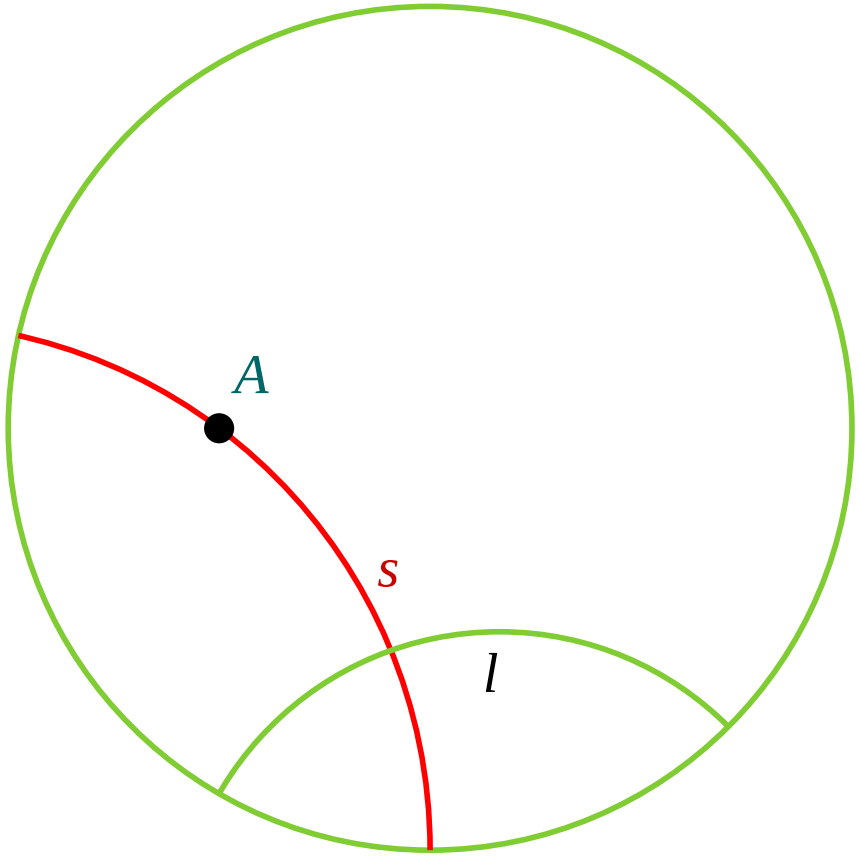
Poincaré's disc model

a point A
a line ℓ



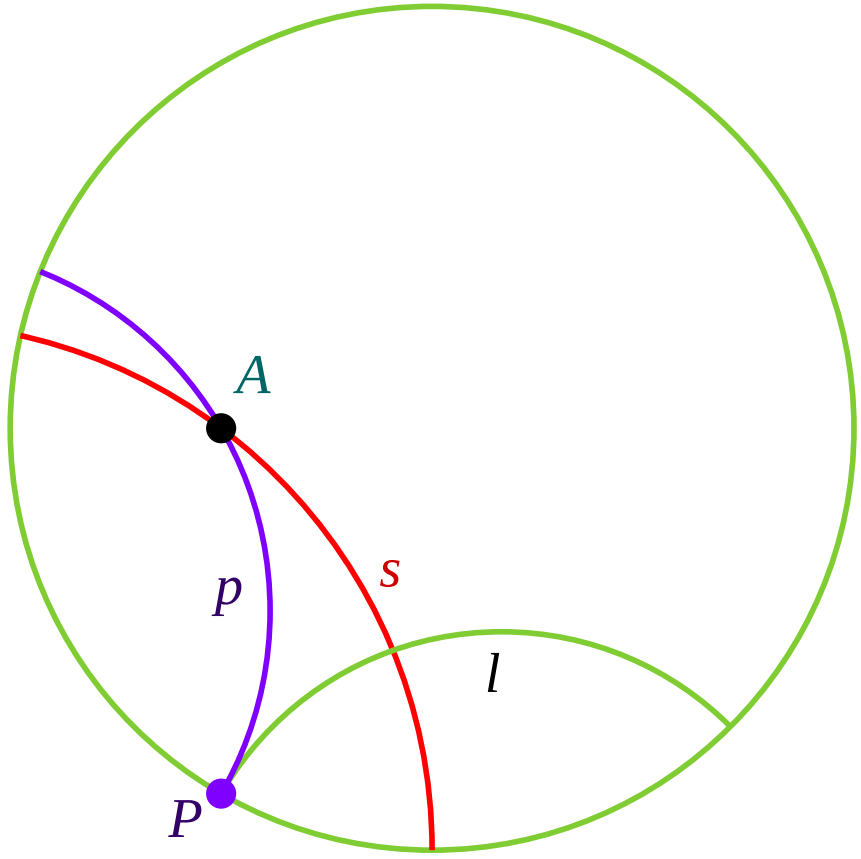
Poincaré's disc model

a **secant**
through A
which cuts ℓ



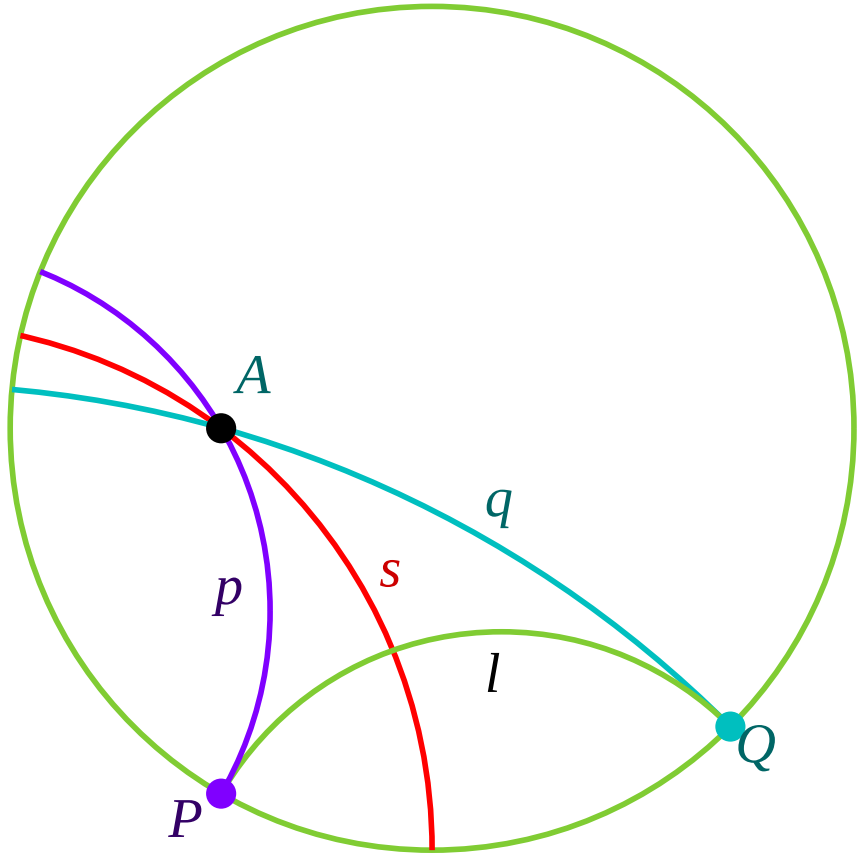
Poincaré's disc model

a **parallel** p
to ℓ
through A



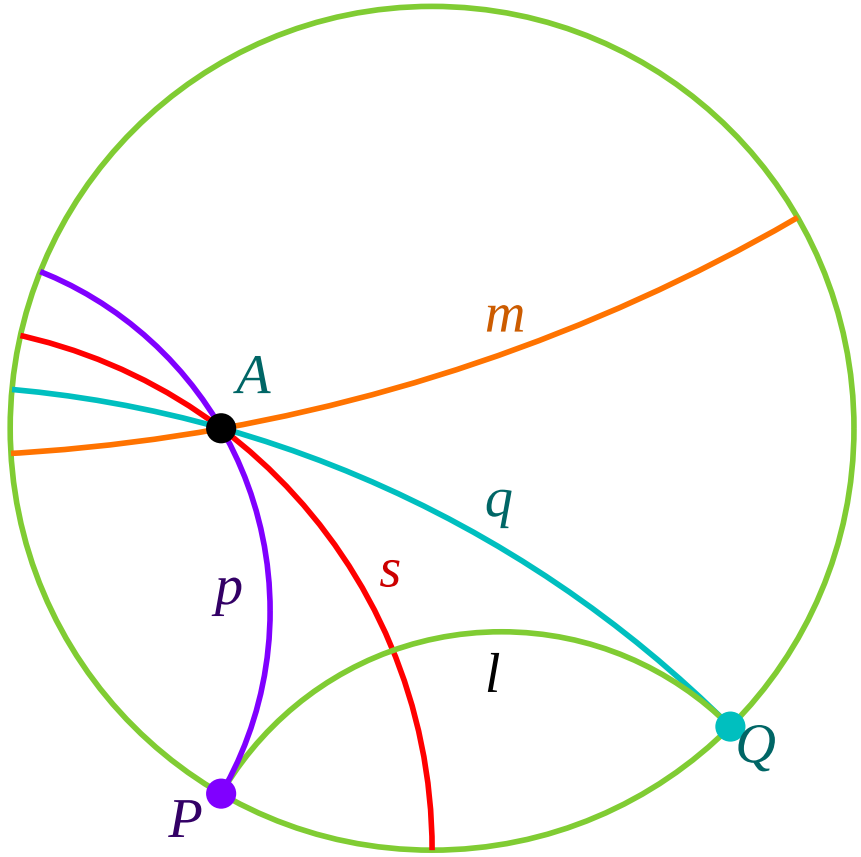
Poincaré's disc model

another
parallel q
to ℓ
through A



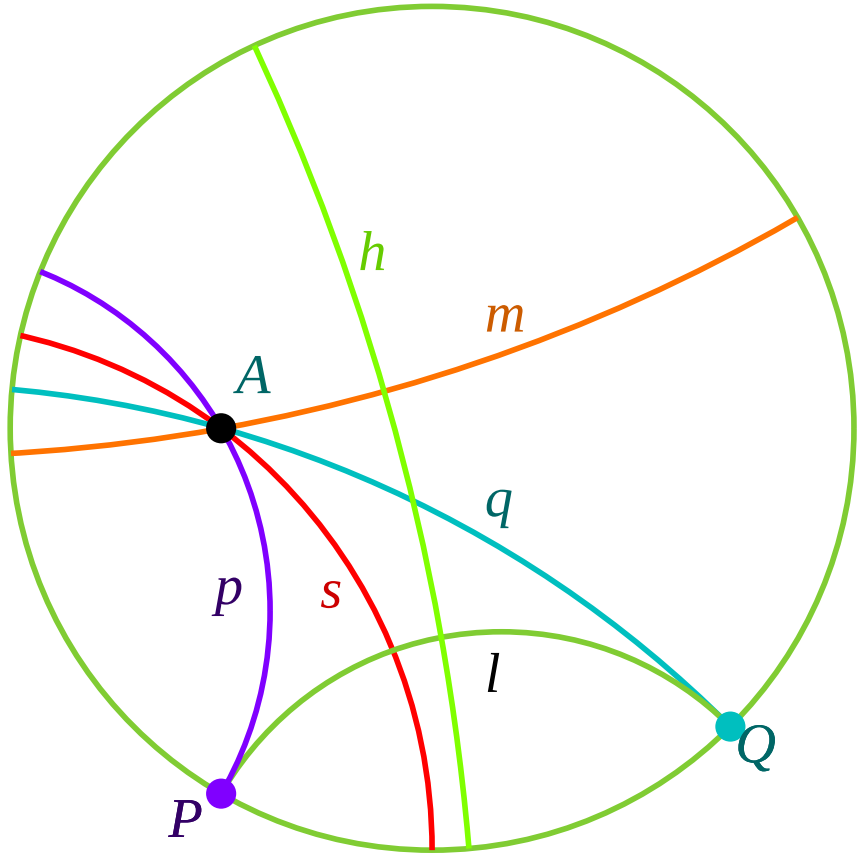
Poincaré's disc model

a **non secant** line m to ℓ through A



Poincaré's disc model

the **common perpendicular** to ℓ and to m



tilings of the hyperbolic plane

there are infinitely many
tessellations in the
hyperbolic plane
thanks to **Poincaré's theorem**

tilings of the hyperbolic plane

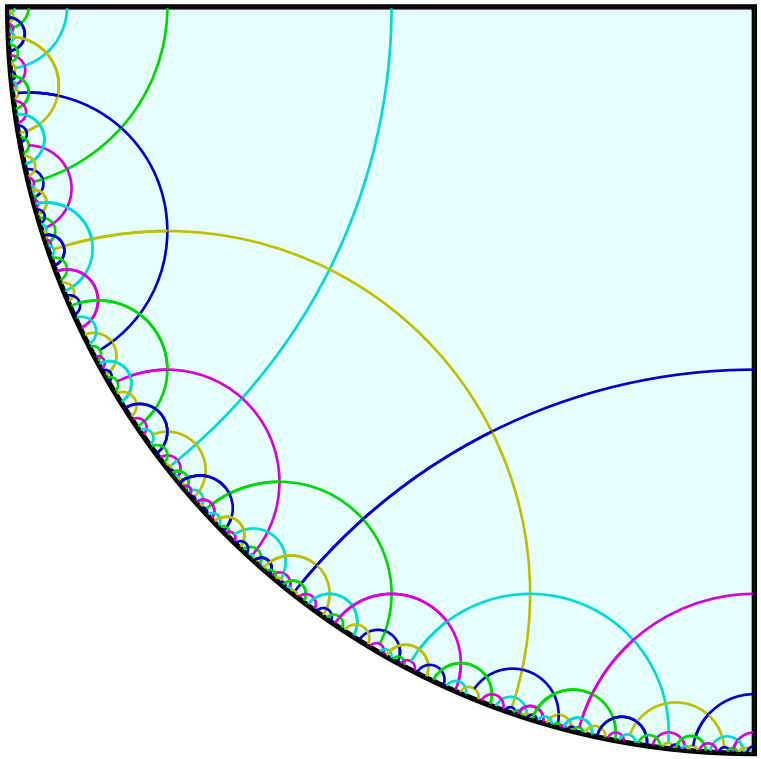
among them, very special objects:

the **infinigons**

infinigons

in the south-western corner:

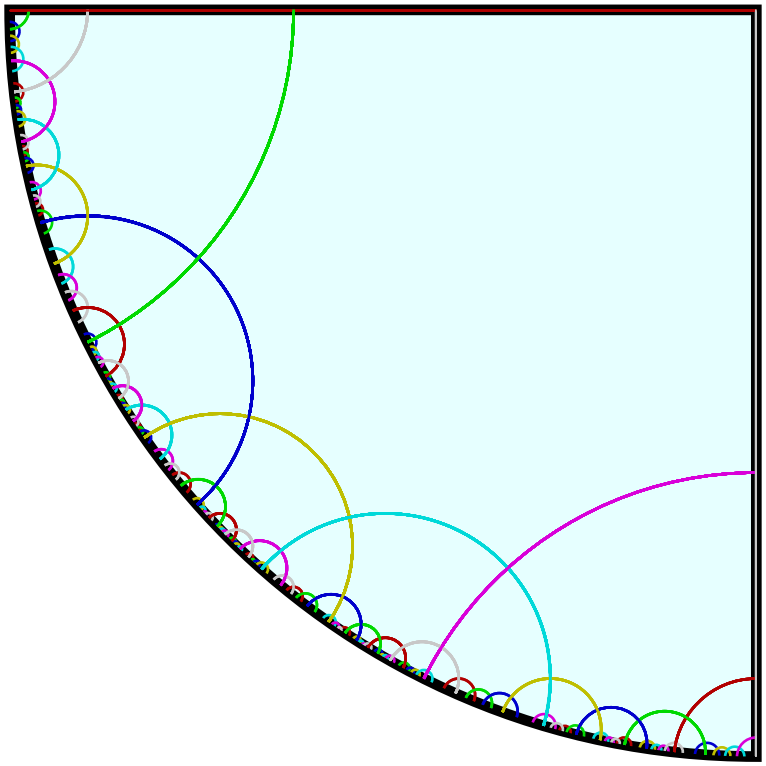
the pentagrid:



infinigons

in the south-western corner:

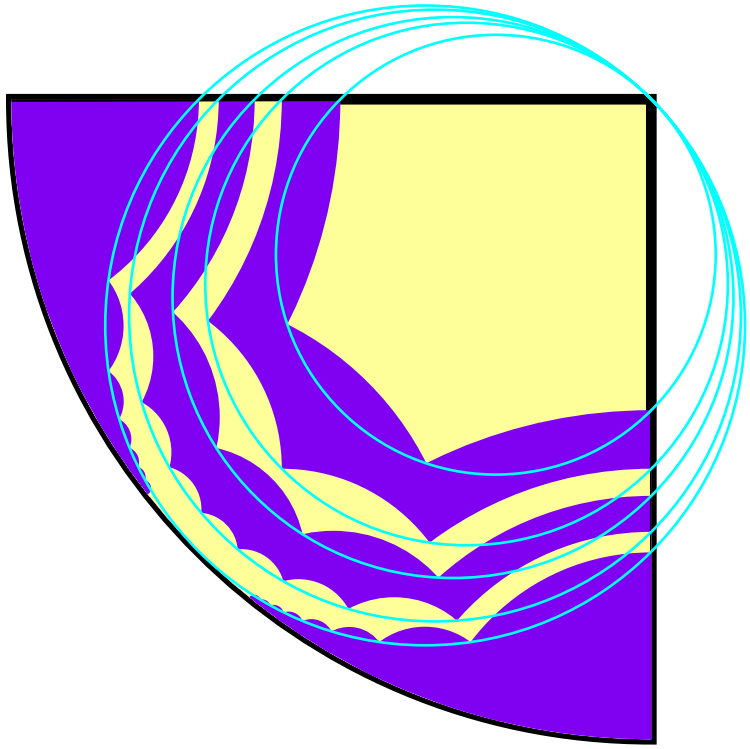
the rectangular heptagrid:



infinigons

in the south-western corner:

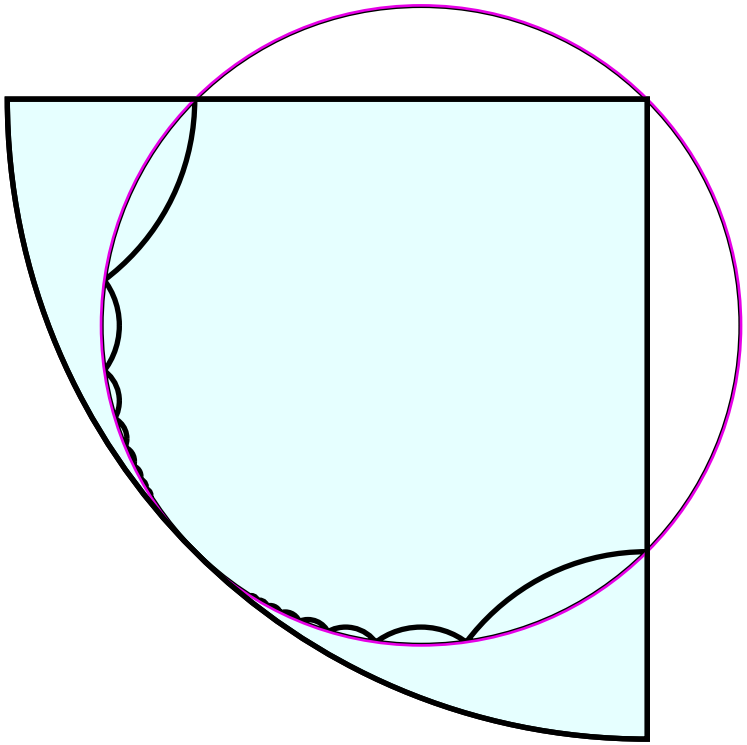
going to the limit:



infinigons

in the south-western corner:

at the limit:

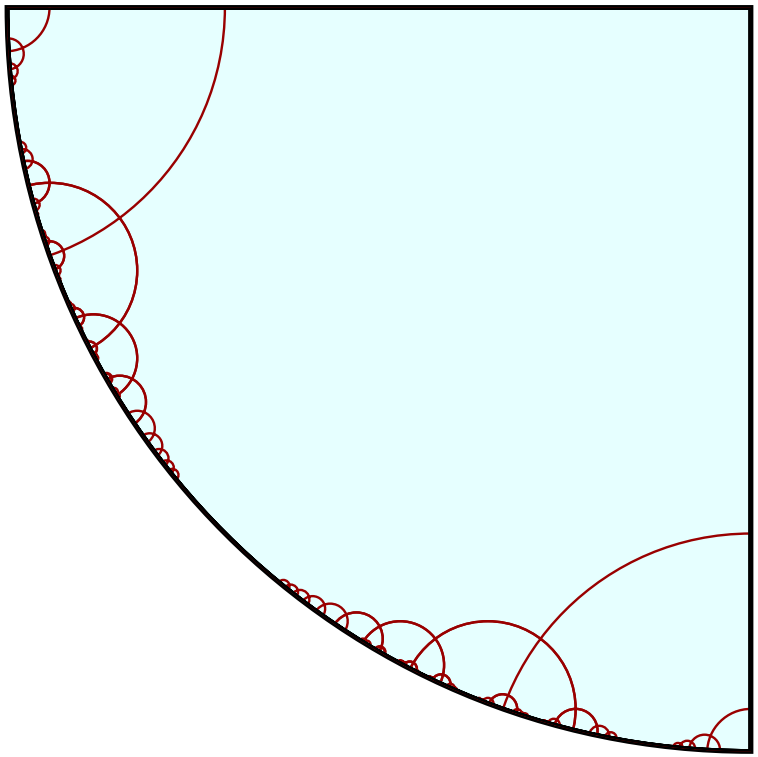


infinigons and infinigrids

now, the infinigons may pave the

hyperbolic plane,

Coxeter, Rosenfeld, Margenstern

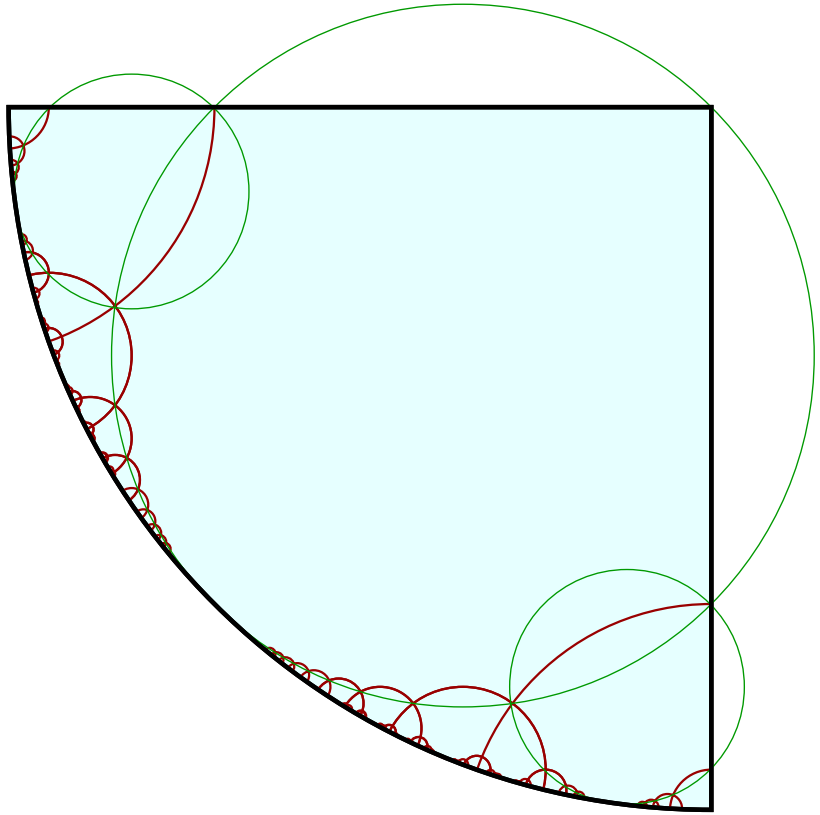


infinigons and infinigrids

now, the infinigons may pave the

hyperbolic plane,

Coxeter, Rosenfeld, Margenstern



infinigons and infinigrids

specific CA's on the infinigrid,

Margenstern-Grigorieff, 2002:

infinitely many neighbours but finite information

solution: an **accurate look**:

a cell knows the set of states taken by **all** its neighbours but it is not able to locate which of them has such a state

infinigons and infinigrids

in this model, **any formula of the arithmetical hierarchy can be decided in linear time by an appropriate such CA**

appending two integer-valued registers with one read-only and the possiblity to perform the basic arithmetic operations, then **there is such a CA able to decide any formula of the arithmetical hierarchy in linear time with respect to the length of the formula**

4. Computer science: a revolution

Computer science: a revolution

4.1 how big is the revolution?

4.2 philosophical questions

4.2.a initialization again

4.2.b space and time

4.2.c randomness

4.2.d artificial intelligence

Computer science: a revolution

how big is the revolution?

usually accepted:

the invention of **printing**

my claim:

at least as that of **writing**

Computer science: a revolution

arguments:

power of diffusion or printing outpaced

new possiblities for creation

clear in music, painting, image management

also clear in writing:

revolution of the hypertext

Computer science: a revolution

for science:

extraordinary tool for exploration

writing introduced a definite help for memory

the computer goes much further

things that can control themselves:

autonomous behaviour

Computer science: a revolution

for every day life:

mobile phone:
in 1990: almost none
nowadays: billions of devices

personal computers:
somewhere above one billion

official use by administrations:
example: for paying taxes

Computer science: a revolution

the revolution started

just a few years ago

we are at the beginning only

philosophical questions

initialization

outside metaphysical aspect,

what is outside of theory
always exists

role of the observer

philosophical questions

time and space

already seen: **discrete** versus **continuous**

computer science describes a **discrete** world

giving tools to explore it:

direct simulation by discrete objects:

cellular automata, multi-agent models

philosophical questions

time

time in computer science:

discrete and **irreversible**

possibly infinite, even in both directions:

best modelization of an operating system

philosophical questions

irreversible time

in the notion of an algorithm:

before an instruction and **after**

note asymmetrical notation:

$$a := b$$

other irreversible phenomena:

one way functions
reverse engineering

philosophical questions

causality,

exotic models show:

laws of physics:
- **constraint** laws
- not at all **determining** laws:

partial differential equations
do not explain the work of a computer

computer have their own laws

the same for many other fields

philosophical questions

randomness

algorithmic theory of information

Chaitin, Kolmogorov

random finite sequence S:
the shortest way to give S,
copy it!

philosophical questions

what is thinking?

Turing gives no definition
he proposes the **imitation game**:

A: a man, B: a machine
C, a man, knows them as X, Y
but ignoring who is A
C communicates with A and B
through typed papers only

philosophical questions

what is thinking?

the fly metaphor:

IA is to man's thinking
what planes are to birds

important focus:
sequential processes versus parallelism

philosophical questions

sequential/parallel

artificial thinking:

big success in sequential algorithms

parallelism: big problem

no satisfactory model

many of them:

- collaboration, competition
- synchrony, asynchrony

another aspect of the role of **time**

philosophical questions

proofs and demonstrations

formal proofs:

allow mechanical checking

mathematical proofs:

persuasion, convinction
intersubjective agreement

philosophical questions

why this difference?

formal proof theorey says:

cut elimination:

cut: $$\frac{\Gamma \vdash A, \Delta \qquad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$$

given a formal proof,
possible to get an equivalent proof with no cut

philosophical questions

importance of cuts:

formal proofs:
cut-free proofs easy to check
but very long

formal structure of natural proofs:

many cuts and substitutions

philosophical questions

many other aspects,

for example, **complexity theory**

all of them point at how large
is the world opened for us
by Turing's discoveries

Thank you
for your attention!