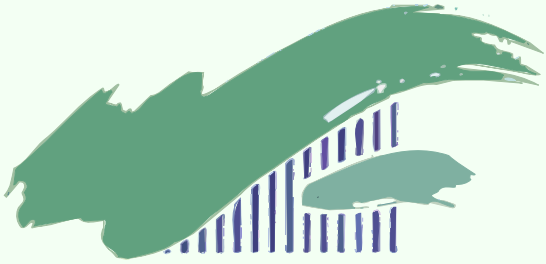


UNIVERSITE D'ORLEANS
Faculté des Sciences
LIFO

Laboratoire d'Informatique Fondamentale d'Orléans
4, rue Léonard de Vinci, BP 6759
F-45067 Orléans Cedex 2
FRANCE

Rapport de Recherche

www : http://www.univ-orleans.fr/SCIENCES/LIFO/

Abstract geometrical computation 1 : embedding Black hole computations with rational numbers

Jérôme Durand-Lose
Université d'Orléans, LIFO

Rapport N^o **2005-05**

Abstract geometrical computation 1: embedding Black hole computations with rational numbers*

Jérôme Durand-Lose†
Laboratoire d'Informatique Fondamentale d'Orléans,
Université d'Orléans,
B.P. 6759, F-45067 Orléans Cedex 2.
jerome.durand-lose@univ-orleans.fr

December 2, 2005

Abstract

The Black hole model of computation provides super-Turing computing power since it offers the possibility to decide in finite (observer's) time any recursively enumerable ($\mathcal{R}.\mathcal{E}.$) problem. In this paper, we provide a geometric model of computation, *conservative abstract geometrical computation*, that, although being based on rational numbers (and not real numbers), has the same property: it can simulate any Turing machine and can decide any $\mathcal{R}.\mathcal{E}.$ problem through the creation of an accumulation. Finitely many signals can leave any accumulation, and it can be known whether anything leaves. This corresponds to a black hole effect.

key-words

Abstract geometrical computation, Black hole model, Energy conservation, Malament-Hogarth space-time, Super-Turing computation, Turing universality, Zeno phenomena.

Disclaimer: None of the physicist aspects of this paper is to be considered as definitely true. The author, being a computer scientist with little knowledge on the matter, would not feel insulted if one would consider these mere inventions/illusions. However, we do not pretend to explain or describe black holes, but just to provide a computer scientist insight and model mostly directed to the computer science community. This paper could have been presented as a model of computation with special features, but since so much similarities exist, we stress on the correspondence with the Black hole model.

*An extended abstract version of this paper (proofs were only schetched) was published in [DL05].

†This research was partly done while the author was member of LIP, ÉNS Lyon and of Université de Nice-Sophia Antipolis, France.

1 Introduction

Theoretical physicists picture the universe as a gigantic information processor/computer. They address the limits of the Church-Turing thesis as they get insights of possible space-times abiding Einstein's equations but providing super-Turing computing power [Hog94, LN04]. The idea is to have the possibility to use an infinite amount of time on a separate future endless curve to try solving a recursively enumerable ($\mathcal{R}.\mathcal{E}.$) problem, such that the result, or the absence of any result, can be retrieved in finite time in the main curve. For the theoretical computer scientist, this is related to infinite Turing computation or computation on ordinals [HL00, Ham02].

Malament-Hogarth space-times [Hog00, EN02] provides this. Roughly speaking, the idea is the following. General relativity permits space-times in which time runs with different "speeds" in different regions. The life-lines of the computer and the observer are arranged in such a way that the machine has an infinite amount of time ahead of it; but any signal it returns is received by the observer within a bounded local delay (measured on the observer's clock). After this finite delay, the observer knows whether the computation ever stopped (by noticing whether anything was received) and what the answer is. It is thus possible to decide any $\mathcal{R}.\mathcal{E}.$ problem in "finite time".

We present a new computing model of computation that has the same capability through a similar approach: *Abstract geometrical computation*. This model considers Euclidean lines. The support of space and time is thus $\mathbb{R}$. Computations are produced by *signal machines* which are defined by a finite set of *meta-signals* and a finite set of *collision rules*. Signals are atomic information, corresponding to meta-signals, moving at constant speed thus generating Euclidean line segments on space-time diagrams. Collision rules are pairs *(incoming meta-signals, outgoing meta-signals)*, that define a mapping (which means determinism) over sets of meta-signals. They define what happens when signals meet, *i.e.* at the extremities of the line segments.

A configuration (at a given time or the restriction of the space-time diagram to a given time) is a mapping from $\mathbb{R}$ to meta-signals, collision rules, and two special values: void (*i.e.* nothing there) and accumulations (amounting for black holes). There should be finitely many positions not mapped to void. The time scale is $\mathbb{R}^+$, since time is continuous, there is no such thing as a "next configuration". The following configurations are defined by the uniform movement of each signal, the speed of which is defined by its associated meta-signal. When two or more signals meet, this produces a *collision* defined by a collision rule. In the configurations following a collision, incoming signals are removed and outgoing signals corresponding to the outgoing meta-signals are added.

Zeno like acceleration and accumulation can be constructed as on the left of Fig. 2. This provides the black hole-like artifact for deciding $\mathcal{R}.\mathcal{E}.$ problems. But accumulations can lead to an uncontrolled burst of signals producing infinitely many signals in finite time (as in Fig. 3). In order to avoid this, we impose a *conservativeness* condition on the rules: a positive energy is defined for every meta-signal, the sum of these energies must be conserved by each rule. Thus no energy creation is possible, and the number of signals is bounded.

Each signal corresponds to a meta-signal which indicates the slope of its trace on the space-time diagram. Since there are finitely many meta-signals, there are finitely many slopes. This limitation may seem restrictive and unrealistic, even awkward as a

quantification inside an analog model of computation. Let us notice that, first, it comes from cellular automata (CA) (as explained below): once a discrete line is identified, wherever (and whenever) the same pattern appears, the same line is expected, thus with the same slope. Second, we give two pragmatic arguments: (1) laws to compute new slopes in collisions are not so easy to design and pretty cumbersome to manipulate; (2) there is already much computing power.

Abstract geometrical computation comes from the common use, in the literature on CA, of Euclidean lines to model discrete lines in (discrete) space-time diagrams of CA to access dynamics or to design. Cellular automata form a well known and studied model of computation and simulation. Configurations are $\mathbb{Z}$-arrays of cells the states of which belong to a finite set. Each cell can only access the states of its neighboring cells. All cells are updated iteratively and simultaneously. The main characteristics of CA, as well as abstract geometrical computation, are: parallelism, synchronicity, uniformity and locality of updating. The space-time diagrams of CA are colorings of $\mathbb{Z} \times \mathbb{N}$ with states as on the left part of Fig. 1. Discrete lines are often observed on these diagrams and idealized as continuous lines as on the right part of Fig. 1. They can be the keys to understanding the dynamics and correspond to so-called *particles*, *soliton* or *signals* as in, *e.g.*, [Ila01, pp. 87–94] or [BNR91, HSC01, JSS02, Siw01]. They can also be the tool to design CA for precise purposes and then named *signals* and used for, *e.g.*, prime number generation [Fis65], Turing machine simulation [LN90], firing squad synchronization [Got66, VMP70, Maz96b] or reversible simulation [DL97]. These discrete line systems have also been studied on their own [Maz96a, MT99, DM02]. All these papers, and many more, implicitly use abstract geometrical computation.

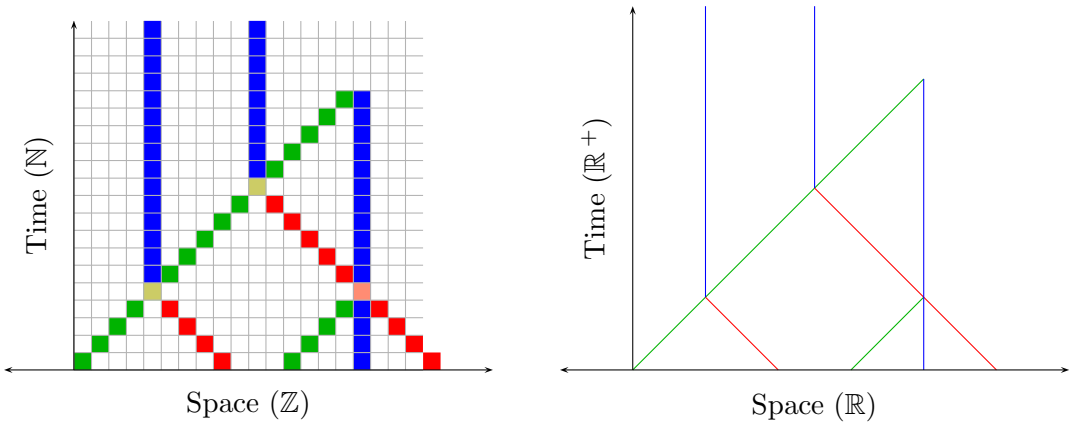


Figure 1: Space-time diagram of a cellular automaton and its signal machine counterpart.

Before presenting our results, we want to convince the reader that it is not just "one more model of computation". First, it does not come "out of the blue" because of its CA origin. Second, to our knowledge[1], it is the only model that is a dynamical system with continuous time and space but finitely many local values. The closest model we know of is the Mondrian automata of Jacopini and Sontacchi [JS90]. Their space-time diagrams are mappings from $\mathbb{R}^n$ to a finite set of colors. They should be bounded finite polyhedra; we are only addressing lines –faces are not considered– and our diagrams may be unbounded and accumulation may happen (they just forbid it). Another close model

[1] A brief tour of analog/super-Turing models of computation can be found in [DL03, Chap. 2].

is the piecewise-constant derivative system [AM95, Bou99]: $\mathbb{R}^n$ is partitioned into finitely many polygonal regions; the trajectory is defined by a constant derivative on each region, thus an orbit is a sequence (possibly over an ordinal) of (Euclidean) line segments. This model is sequential –there is only one "signal"– and the faces that delimit the regions are artifacts that do not exist in our model. Nevertheless, it also uses accumulations to decide $\mathcal{R}.\mathcal{E}.$ problems.

In this paper, space and time are restricted to rational numbers. This is possible since all the operations used preserve rationality. All quantifiers and intervals should be understood over $\mathbb{Q}$, not $\mathbb{R}$. Extending the definitions to real values is automatic but only the rational case is addressed here. Let us note that rational numbers can be implemented exactly on a computer while this is not possible for real numbers.

After formally defining our model in Sect. 2, we prove that any Turing-computation can be carried out through the simulation of 2-counter automata in Sect. 3. The values of the counters are encoded by positions (fixed signals indicates the scale) and the instructions are going forth and back between them. The continuous nature of space is used here: all $\{1/2^n\}_{n\in\mathbb{N}}$ positions do exist.

In Sect. 4, we show how to modify a signal machine so that it can dynamically scale by 1/2 the rest of the computation. With an automatic and infinite iteration of this shrinking process, it is possible to bound temporally a computation that is already spatially bounded. This method is constructive and relies on the assumption that space and time are continuous. The construction generates an accumulation.

We explain how to use this accumulation for deciding $\mathcal{R}.\mathcal{E}.$ problems in Sect. 5. Conclusion, remarks and perspectives are gathered in Sect. 6.

2 Definitions

Abstract geometrical computations are defined by the following machines:

Definition 1 A *signal machine* is defined by (M, S, R) where M (*meta-signals*) is a finite set, S (*speeds*) is a mapping from M to $\mathbb{Q}$, and R (*collision rules*) is a partial mapping from the subsets of M of cardinality at least 2 into the subsets of M (sets in both domain and range must not contains two elements of identical speeds).

The elements of M are called *meta-signals*. Each instance of a meta-signal is a *signal* which corresponds to a line segment in the space-time diagram. The mapping S assigns rational *speeds* to meta-signals, *i.e.* the slopes of the segments. The *collision rules*, denoted $\rho^- \to \rho^+$, define what happens when two or more signals meet. It also defines the extremities of the segments. Since R is a mapping, the signal machines are deterministic.

The *extended value set*, V, is the union of M and R plus two symbols: one for void, $\oslash$, and one for an accumulation (or black hole) ❋. A *configuration*, c, is a total mapping from $\mathbb{Q}$ to V such that the set $\{\, x \in \mathbb{Q} \mid c(x) \neq \oslash \,\}$ is finite.

A signal corresponding to a meta-signal μ at a position x, *i.e.* $c(x) = \mu$, is moving uniformly with constant speed $S(\mu)$. A signal must start in the initial configuration or be generated by a collision. It must end in a collision or in the last configuration. This corresponds to condition 2. in Def. 2.

At a $\rho^- \to \rho^+$ collision, all, and only, signals corresponding to the meta-signals in ρ^- (resp. ρ^+) must end (resp. start). No other signal should be present. This corresponds to condition 3. in Def. 2. A black hole corresponds to an accumulation of collisions and disappears without a trace. This corresponds to condition 4. in Def. 2.

Let S_{min} and S_{max} be the minimal and maximal speeds (*i.e.* the extrema of S). The *causal past*, or *light-cone*, arriving at position x and time t, $J^-(x,t)$, is defined by all the positions that might influence the information at (x,t) through signals, formally:

$$J^-(x,t) = \{\ (x',t') \mid (0 \leq S_{max}(t-t') - x + x') \wedge (0 \leq x - x' - S_{min}(t-t'))\ \}\ .$$

Before formally defining the dynamics by space-time diagrams, we want to point out the black hole formation example of Fig. 2. This example is so simple (*i.e.* 4 meta-signals and 2 collision rules) that such a situation cannot be excluded.

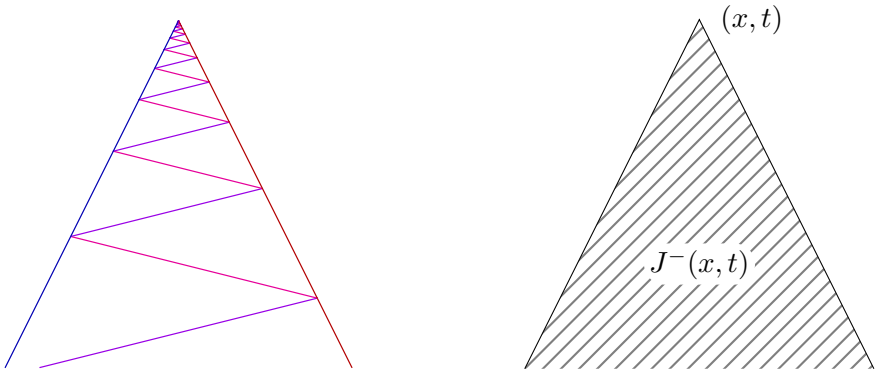


Figure 2: Black hole and light-cone.

Definition 2 The *space-time diagram*, or orbit, issued from an initial configuration c_0 and lasting for T, is a mapping c from $[0,T]$ to configurations (*i.e.* a mapping from $\mathbb{Q} \times [0,T]$ to V) such that, $\forall (x,t) \in \mathbb{Q} \times [0,T]$:

1. $\forall t \in [0,T]$, $\{\, x \in \mathbb{Q} \,|\, c_t(x) \neq \oslash \,\}$ is finite,
2. if $c_t(x) = \mu$ then $\exists t_i, t_f \in [0,T]$ with $t_i < t < t_f$ or $0 = t_i = t < t_f$ or $t_i < t = t_f = T$ s.t.:
   - $\forall t' \in (t_i, t_f)$, $c_{t'}(x + S(\mu)(t'-t)) = \mu$,
   - $t_i = 0$ or $c_{t_i}(x_i) \in R$ and $\mu \in (c_{t_i}(x_i))^+$ where $x_i = x + S(\mu)(t_i - t)$,
   - $t_f = T$ or $c_{t_f}(x_f) \in R$ and $\mu \in (c_{t_f}(x_f))^-$ where $x_f = x + S(\mu)(t_f - t)$;
3. if $c_t(x) = \rho^- \to \rho^+ \in R$ then $\exists \varepsilon$, $0 < \varepsilon$, $\forall t' \in [t-\varepsilon, t+\varepsilon] \cap [0,T]$, $\forall x' \in [x-\varepsilon, x+\varepsilon]$,
   - $c_{t'}(x') \in \rho^- \cup \rho^+ \cup \{\oslash\}$,
   - $\forall \mu \in M$, $c_{t'}(x') = \mu \Rightarrow \bigvee \left\{ \begin{array}{l} \mu \in \rho^- \text{ and } t' < t \text{ and } x' = x + S(\mu)(t'-t)) \\ \mu \in \rho^+ \text{ and } t < t' \text{ and } x' = x + S(\mu)(t'-t)) \end{array} \right.$;
4. if $c_t(x) = \text{❋}$ then
   - $\exists \varepsilon > 0$, $\forall (x',t') \notin J^-(x,t)$, ($|x-x'| < \varepsilon$ and $|t-t'| < \varepsilon$) $\Rightarrow c_{t'}(x) = \oslash$,
   - $\forall \varepsilon > 0$, $\{\ (x',t') \in J^-(x,t) \mid t - \varepsilon < t' < t \wedge c_{t'}(x') \in R\ \}$ is infinite.

On space-time diagrams, the traces of signals represent line segments whose directions are defined by $(S(.), 1)$ (1 is the temporal coordinate). Collisions correspond to the extremities of these segments. Examples of space-time diagrams are provided by the various figures. Time is always increasing upwards. This definition can easily be extended to the $T = \infty$ case.

The three space-time diagrams of Fig. 3 provide examples of possible but unwanted cases. They are not compatible with Def. 2 if the time at the accumulation is to be considered. In each case, the number of signals is bursting to infinity and black holes are not isolated. This is unwanted because on the one hand it corresponds to the free apparition of energy, and on the other hand we fell that black holes should be dimensionless points. The other space-time diagrams of Fig. 3 provide even more unwanted cases: in the center one, they accumulate on a Cantor set; and in the right one, there is an accumulation of signals that forms a everlasting discontinuity. To prevent this as well as many others, we introduce the following restriction. It corresponds to the conservation of some energy.

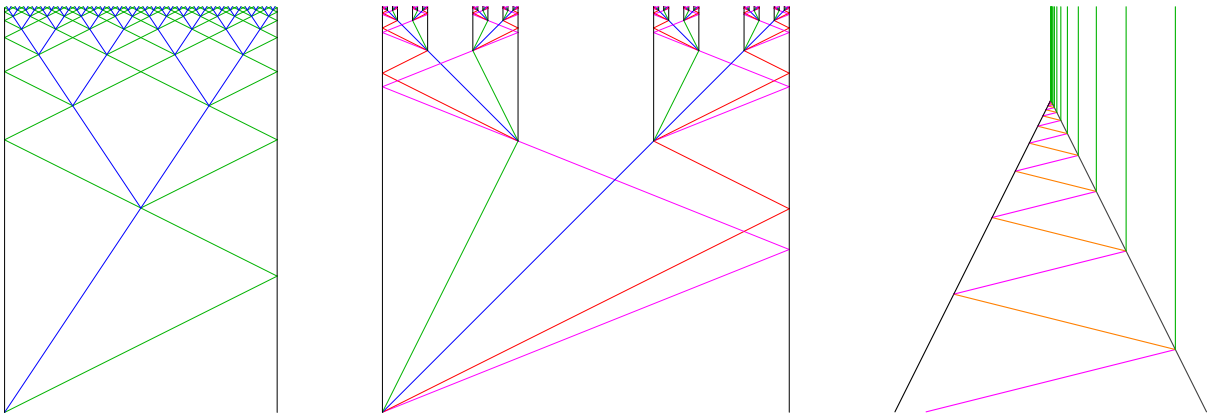


Figure 3: Unwanted phenomena.

Definition 3 A signal machine is *conservative* when an atomic positive energy is defined for all meta-signals ($E : M \to \mathbb{N}^*$) such that the total energy of the system is preserved, *i.e.* the sum of all the energy of existing signals is a constant of the system. This is equivalent to accept only rules that preserve this energy, *i.e.* the sum of the energy of incoming meta-signals equals the sum of outgoing ones.

Allowing only integer values for the energy is not restrictive since there are finitely many meta-signals and rules.

Conservativeness is straightforward if the condition on rules is satisfied. If it is not satisfied, it is very easy to built a configuration such that the only rule involved is not conservative and then the energy is not preserved.

Property 4 *Given a conservative signal machine and an initial configuration, the number of signal in any following configuration, as well as the number of accumulations, is bounded (by the total energy divided by the least atomic energy).*

Energy can only be lost in "black hole" formation, *i.e.* accumulation. A sub-case of conservativeness is when all the meta-signals have the same energy and the number of

in and out meta-signals are always equal. This is the case in the rest of this paper. We chose to present a more complex notion since it is much less restrictive and better suits the notion of the energy conservation.

3 Turing-computation capability

In this section, we deal with the computing power of signal machines and prove the following:

Theorem 5 *Conservative signal machines are able to carry out any Turing-computation.*

The demonstration is carried out through the simulation of any 2-counter automaton. A 2-counter automaton is a finite automaton coupled with two counters, A and B. The possible actions on any counter are *add/subtract* 1 and *branch if non-zero*. These machines can be described with a six-operations (the three aforementioned ones for each of the two counters) assembly language with branching labels as on the left part of Fig. 9 (see [Min67] for more on 2-counter automata).

3.1 Construction

The simulation is carried out with both counters encoded by relative positions according to two fixed signals zero and one. These two signals form a scale on the diagram. The counter A (resp. B) is encoded by a single signal a (b) at position $x_A 2^{-a}$ ($x_B 2^{-b}$) as in Fig. 4. The parameter x_A and x_B are rational numbers such that

$$1 < x_A < x_B < 2 \ ; \tag{1}$$

this ensures that the signal a (b) is between zero and one unless its value is zero and in such a case it is on the other side of one. Let us note that the values of x_A and x_B prevent the signals from occupying the same place and from being on the scale signals. As can be easily checked on the constructions in the rest of this section, they also prevent that any collision happens with an unconcerned signal.

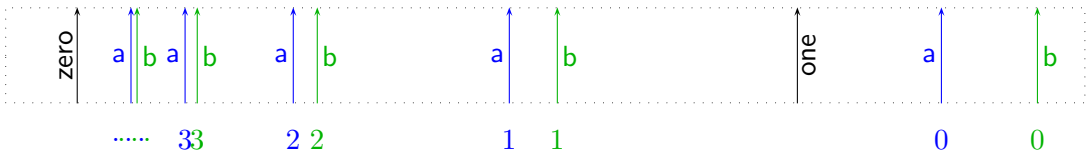


Figure 4: Encoding positions of counters.

The current instruction (*e.g.* n) is encoded as the signal $\overleftarrow{\mathsf{n}}$. It moves back to zero, bounces, carries out the operation and returns as the next operation. The five possible configurations are given in Fig. 5.

The fact that a signal encoding a counter is on the other side of one only for the value 0 provides an easy way to test whether the counter is zero for branching or subtracting 1: going rightward one is encountered first if and only if the value of the counter is 0.

There are two kinds of meta-signals: 8 for the counters and borders, and the ones generated for the program. The meta-signals of the first kind are: zero, one, a and b of

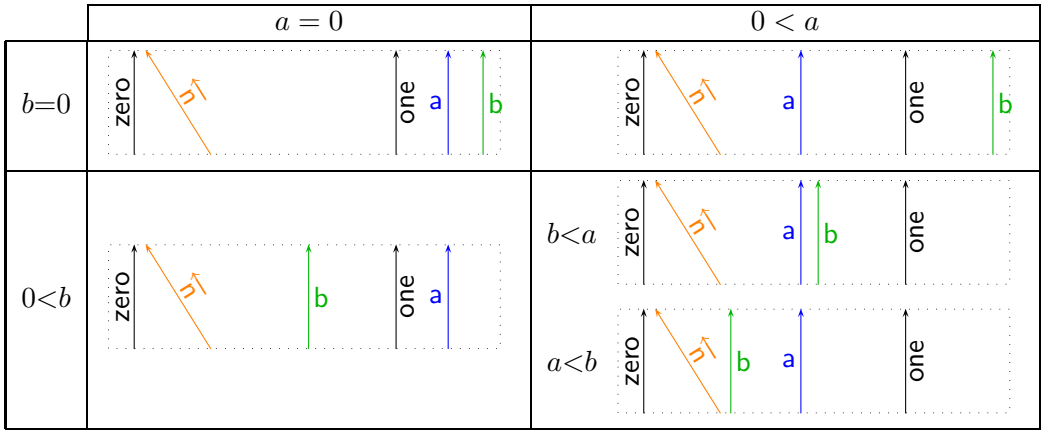


Figure 5: Encoding of configurations.

speed 0 used to mark the borders and to encode A and B, and $\overleftarrow{\mathsf{a}}$ ($\overleftarrow{\mathsf{b}}$) and $\overrightarrow{\mathsf{a}}$ ($\overrightarrow{\mathsf{b}}$) of speed -1 and 1 used to increment/decrement A (B). For the second kind, each line n of the program is converted into $\overrightarrow{\mathsf{n}}$ and $\overleftarrow{\mathsf{n}}$ of speed 2 and -2, and possibly $\overrightarrow{\mathsf{n'}}$ and $\overleftarrow{\mathsf{n'}}$ of speed 3 and -3 to carry out increment and decrement as explained below. In the first kind, we distinguish the a *family* $\{\mathsf{a}, \overleftarrow{\mathsf{a}}, \overrightarrow{\mathsf{a}}\}$ and the b *family* $\{\mathsf{b}, \overleftarrow{\mathsf{b}}, \overrightarrow{\mathsf{b}}\}$. We call the second kind the *instruction family*.

First, any instruction bounces on zero to be on the left of any other signal and thus be in the right position to start carrying out any instruction. This is achieved by the following rule:

$$\{\mathsf{zero}, \overleftarrow{\mathsf{n}}\} \quad \rightarrow \quad \{\mathsf{zero}, \overrightarrow{\mathsf{n}}\}.$$

The full transformation of a program into a signal machine is given in Fig. 8. We only detail the collision rules generated for the most complicated case: a `A--` instruction (at line n). The rules are the following:

$$\{\overrightarrow{\mathsf{n}}, \mathsf{one}\} \rightarrow \{\overleftarrow{\mathsf{n+1}}, \mathsf{one}\}, \qquad \{\overrightarrow{\mathsf{n}}, \mathsf{a}\} \rightarrow \{\overleftarrow{\mathsf{n'}}, \overrightarrow{\mathsf{a}}\},$$
$$\{\mathsf{zero}, \overleftarrow{\mathsf{n'}}\} \rightarrow \{\mathsf{zero}, \overrightarrow{\mathsf{n'}}\}, \qquad \{\overrightarrow{\mathsf{n'}}, \overrightarrow{\mathsf{a}}\} \rightarrow \{\overleftarrow{\mathsf{n+1}}, \mathsf{a}\}.$$

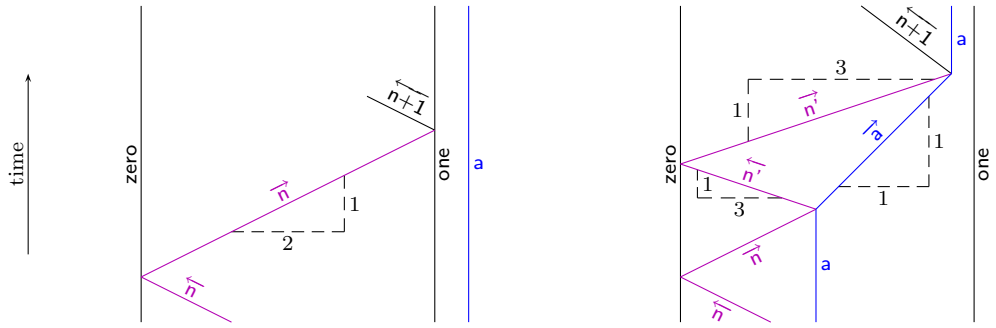
All other rules with $\overrightarrow{\mathsf{n}}$, $\overrightarrow{\mathsf{n'}}$ or $\overleftarrow{\mathsf{n'}}$ are blank, *i.e.*, the same signals are regenerated. The effect of above rules is shown in the space-time diagrams of Fig. 7. The relative position of one and a is very important because a counter already at zero is not decremented. If such is not the case, the distance between zero and a is multiplied by 2 as it can easily be geometrically checked on Fig. 6 where the slopes are indicated with dashed lines.

The instruction `A++` does exactly the same thing but in reverse and the zero case does not have to be considered. The non-zero conditional branching is done by simply noticing that one is met before a if and only if `A` is zero. This is illustrated by the last two space-time diagrams of Fig. 7.

All the instructions on B are carried out similarly. All the rules are detailed in Fig. 8.

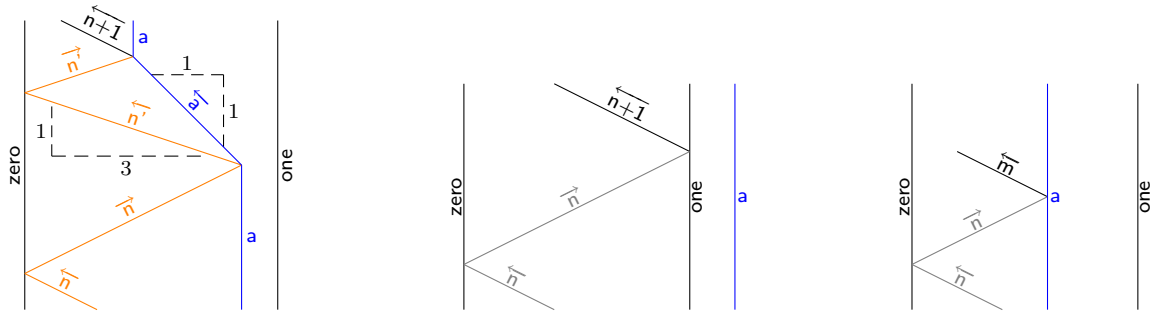
Figure 9 provides the code of a 2-counter automaton and three simulations associated to different initial values. The pictures are strained vertically in order to fit.

The only thing left to consider is the end of the computation, *i.e.* the treatment of the halt. It is not possible to just make the instruction signal disappears since this would yields a non conservative rule. To cope with this, one can choose to let the instruction signal leaves on the left (but this signal could interfere with the rest of the computation),



(a) $a = 0$ case. (b) $0 < a$ case.

Figure 6: Implementation of `A--`.



(a) `A++` (b) $a = 0$ (c) $0 < a$

Figure 7: Implementations of `A++` and n : `IF A!=0` m.

Line number	Instruction	Generated rules
n	any	$\{\text{zero}, \overleftarrow{\text{n}}\} \to \{\text{zero}, \overrightarrow{\text{n}}\}$ $\{\text{zero}, \overleftarrow{\text{n'}}\} \to \{\text{zero}, \overrightarrow{\text{n'}}\}$
n	`A++`	$\{\overrightarrow{\text{n}}, \text{a}\} \to \{\overleftarrow{\text{n'}}, \overleftarrow{\text{a}}\}$ $\{\overrightarrow{\text{n'}}, \overleftarrow{\text{a}}\} \to \{\overleftarrow{\text{n+1}}, \text{a}\}$
n	`B++`	$\{\overrightarrow{\text{n}}, \text{b}\} \to \{\overleftarrow{\text{n'}}, \overleftarrow{\text{b}}\}$ $\{\overrightarrow{\text{n'}}, \overleftarrow{\text{b}}\} \to \{\overleftarrow{\text{n+1}}, \text{b}\}$
n	`A--`	$\{\overrightarrow{\text{n}}, \text{one}\} \to \{\overleftarrow{\text{n+1}}, \text{one}\}$ $\{\overrightarrow{\text{n}}, \text{a}\} \to \{\overleftarrow{\text{n'}}, \overrightarrow{\text{a}}\}$ $\{\overrightarrow{\text{n'}}, \overrightarrow{\text{a}}\} \to \{\overleftarrow{\text{n+1}}, \text{a}\}$
n	`B--`	$\{\overrightarrow{\text{n}}, \text{one}\} \to \{\overleftarrow{\text{n+1}}, \text{one}\}$ $\{\overrightarrow{\text{n}}, \text{b}\} \to \{\overleftarrow{\text{n'}}, \overrightarrow{\text{b}}\}$ $\{\overrightarrow{\text{n'}}, \overrightarrow{\text{b}}\} \to \{\overleftarrow{\text{n+1}}, \text{b}\}$
n	`IF A != 0` m	$\{\overrightarrow{\text{n}}, \text{one}\} \to \{\overleftarrow{\text{n+1}}, \text{one}\}$ $\{\overrightarrow{\text{n}}, \text{a}\} \to \{\overleftarrow{\text{m}}, \text{a}\}$
n	`IF B != 0` m	$\{\overrightarrow{\text{n}}, \text{one}\} \to \{\overleftarrow{\text{n+1}}, \text{one}\}$ $\{\overrightarrow{\text{n}}, \text{b}\} \to \{\overleftarrow{\text{m}}, \text{b}\}$

If $n = N$, replace $\overleftarrow{\text{n+1}}$ by $\overleftarrow{\text{stop}}$. All other rules are blank.

Figure 8: List of all transcriptions.

or to let it bounce indefinitely between zero and one; in both cases, the number of signals is preserved.

3.2 Correction of the simulation

Let the configuration of a 2-counter automaton be denoted (n, a, b) where n is the current line number, a (resp. b) is the value of counter A (resp. B), or (stop, a, b) if the compu-

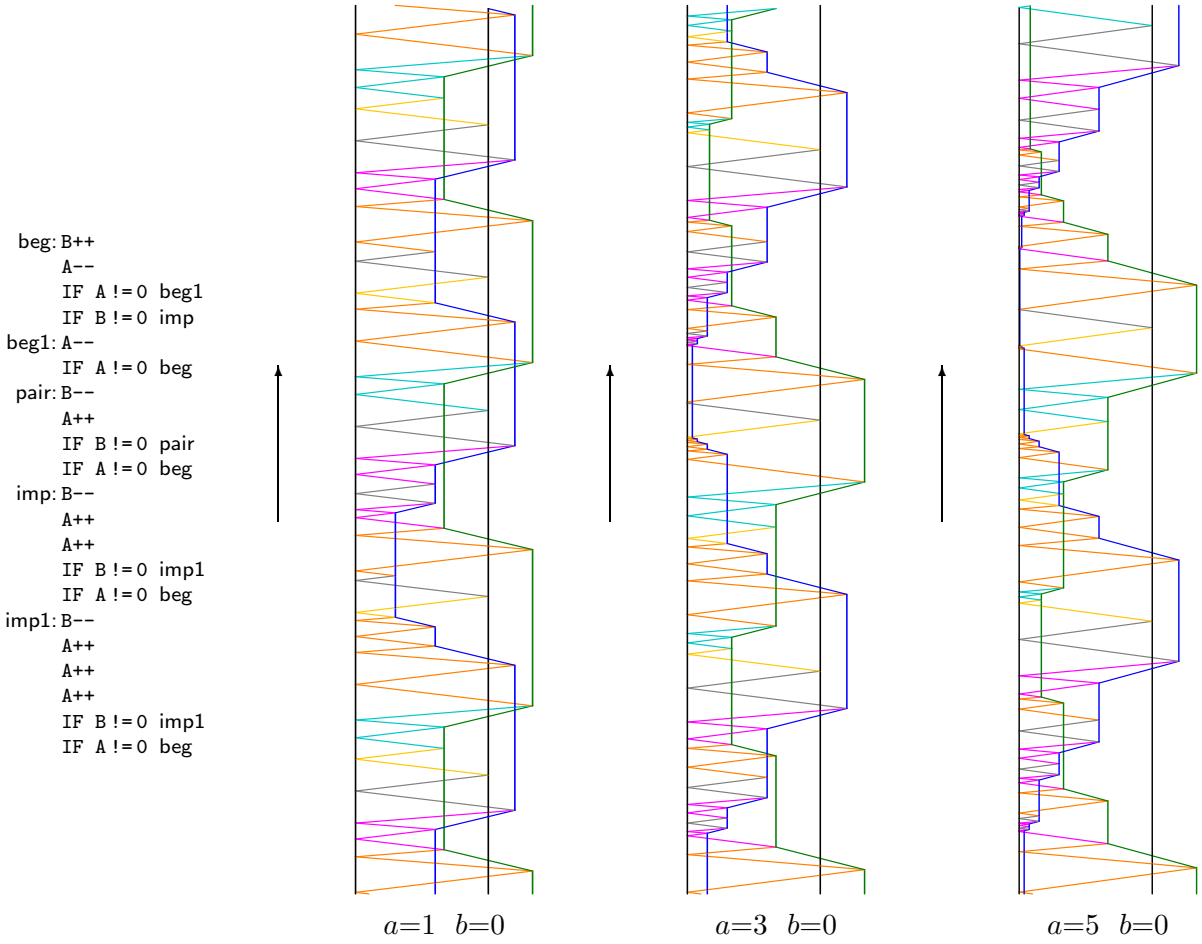


Figure 9: A 2-counter automaton and its simulations for three different initial values.

tation has stopped. To prove the correctness of the simulation, we prove the following:

Lemma 6 *Starting with the configuration* $(1, a_0, b_0)$*, the following properties are verified:*

1. *there is only one* zero *and one* one *signals in each configuration and they are always at the same positions (and theses positions are different and are hereafter used as a scale);*
   there is always exactly one signal from each of the a *family, the* b *family and the instruction family;*
   altogether there are always exactly five signals: one zero *and one* one *signals, and one of each of the three families;*
2. *let us number, starting from* 0*, the collisions of a* $\overleftarrow{\mathsf{n}}$ *signal on* zero *(such collisions are refereed as timing collisions); right before timing collision* i*, the signal machine encodes the configuration of the 2-counter automaton after* i *iterations;*
3. *if the computation of the automaton never stops, there will be infinitely many timing collisions;*
4. *if the computation of the automaton stops in* k *iterations, then the* $k{+}1$*th timing collision generates* $\overleftarrow{\mathsf{stop}}$ *and the final values of the counters are encoded as usual.*

Proof

1. Signals zero and one are always regenerated after a collision and no other collision generates them; so that their numbers remains unchanged. Since their speed is zero, they do not move. In the initial configuration there are exactly one zero and one one signals at different positions. Similarly, a direct inspection of the rules and the initial configuration shows that there is always one and only one signal of each of the a, b and instruction families.
2. Property 2. is proved by induction. It is true at the first timing collision, which is the very first collision, because of the encoding of the initial configuration. It remains to carry the proof for the implementations of the six operations.
   Let first ignore the presence of the unaddressed counter for the three basic operations. Let denote x the (relative to zero and one) position of the signal corresponding to the addressed counter; without loss of generality let assume that it is A.
   If the operation is `IF A != 0` m, then everything happens exactly as in the center or the right part of Fig. 7. The position of the collision is either x or 1. If it is 1, then this means that $1 < x_A 2^{-a}$; since $1<x_A<2$ from (1), $a = 0$; $\overleftarrow{\mathsf{n{+}1}}$ –or $\overleftarrow{\mathsf{stop}}$– is rightfully generated. Similarly, if it is x then $0 < a$; and $\overleftarrow{\mathsf{m}}$ is generated. The value of the counter is unaffected (the signal remains in the same position). The time between the ith and $i{+}1$th timing collision is up bounded by 2.
   If the operation is `A ++`, then everything happens exactly as in the left part of Fig. 7. From the speed of the signals, the positions of the collisions are x, 0 and $x/2$; the final position of a is $x/2$. Since x is $x_A 2^{-a}$, the next position correspond to the next value of the counter: $x_A 2^{-(a+1)}$; the ++ operation was carried out on the counter. The time between the ith and $i{+}1$th timing collision is up bounded by $2x$ and $2x<2x_A<4$ from (1). The right $\overleftarrow{\mathsf{n{+}1}}$ or $\overleftarrow{\mathsf{stop}}$ is generated.
   If the operation is `A --`, then everything happens exactly as in Fig. 6. If the counter holds 0 then $1 < x$ and this is the left scenario. If the counter does not hold 0 then $x < 1$ and this is the right scenario. From the speed of the signals, the positions of the collisions are 1, or x, 0 and $2x$; the final position of a is x or $2x$. Since x is $x_A 2^{-a}$, the next position corresponds to the next value of the counter: $x_A 2^{-(a+1)}$; the -- operation was carried out on the counter. The time between the ith and $i{+}1$th timing collision is up bounded by $2x$ and $2x < 2x_A < 4$. The right $\overleftarrow{\mathsf{n{+}1}}$ or $\overleftarrow{\mathsf{stop}}$ is generated.
   Let us consider now the presence of a signal for the other counter. Above collisions can only take place at coordinate 0, $x/2$, x, $2x$ or 1. From (1), $1<x_A<x_B<2$, if $x = x_A 2^{-a}$ (resp. $x_B 2^{-b}$) then $x_B 2^{-b}$ (resp. $x_A 2^{-a}$) cannot be at any of these location. Thus the other signal does not interfere with the main collisions and only participates in blank collisions, not affecting anything.
3. If the computation of the automaton never stops, then there is also infinitely many timing collisions (the delay between two consecutive timing collisions is up bounded by 4).
4. if the computation of the automaton stops in k iterations the $\overleftarrow{\mathsf{stop}}$ will be normally generated in time less than $4k$.

Q.E.D.

All together, any 2-counter automaton can be simulated by a conservative signal machine; in fact, any finite number of counters can be included and treated similarly.

Signal machines thus form a model of computation which has at least Turing-computing capability.

4 Contraction

4.1 Principle

It is possible to partially strain any space-time diagram as schematized on Fig. 10. The idea is to decompose the upper part according to two non-collinear vectors. One vector is used as a frontier (here the one of speed β). A change of scale is done on the second one (here multiplication by 3 on the axis corresponding to speed α). This is a strain of a given ratio about the second axis. On Fig. 10, the dotted lines indicate how the images of two points are computed. The grey parts indicate the ongoing computation.

This geometrical transformation is easily implemented inside our model: by switching to strained versions of the signals on the frontier, all ongoing computations mimic the unstrained one. The following meta-signals are added: one for the frontier, and one strained meta-signal for each initial meta-signal. All the collision rules are duplicated so that strained signals behave exactly as unstrained ones. Collisions of the form {*frontier and unstrained*}→{*frontier and strained*} are added. New rules are created to account for the possibility of the frontier to pass exactly on a collision. Conservativeness is preserved by setting identical energies to corresponding strained meta-signals.

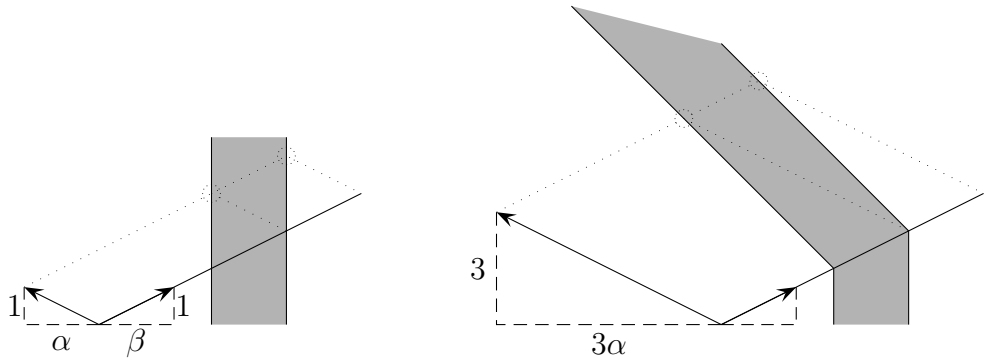


Figure 10: Strain principle.

With this construction, it is possible to build a structure that scales by one half the rest of the computation as illustrated on Fig. 12. The two directions used correspond to v_0 and $4v_0$, where v_0 is big enough. In the left picture, nothing happens. In the middle picture, the lower signal is the frontier and a strain of ratio 1/2 is done about to the upper signal. In the right picture, a second strain takes place: the role of the directions are exchanged, and the ratio is still 1/2. After the two strains, the computation is scaled by 1/2 on both directions, thus on any direction. The whole computation is scaled by 1/2 and the original meta-signals can be used again since the computation undergoes no strain after the second one. This makes it possible to iterate the shrinking, as shall be done after providing and proving the construction for a single shrinking.

4.2 Construction

Let β be the speed of the frontier/toggle signal; the coefficient of the strain is 1 on the direction $(\beta, 1)$. Let α be the other speed and k be the coefficient of the strain on the direction $(\alpha, 1)$. There are some limitations on the possible values of theses three parameters as explained below, at first it is only required that $\alpha \neq \beta$ and $0 < k$.

Let us compute the speed on the strained part, ν', from the speed on the un-strained part, ν:

$$\begin{aligned} (\nu, 1) &= a(\alpha, 1) + b(\beta, 1) \ , \\ \lambda(\nu', 1) &= ka(\alpha, 1) + b(\beta, 1) \ ; \end{aligned} \tag{2}$$

the coefficient λ is needed for normalization. This leads to:

$$\begin{aligned} \lambda(\nu', 1) &= \left(\frac{(k\alpha - \beta)\nu + \alpha\beta(1-k)}{\alpha - \beta}, \frac{(k-1)\nu + \alpha - k\beta}{\alpha - \beta} \right) \ , \\ \lambda &= \frac{(k-1)\nu + \alpha - k\beta}{\alpha - \beta} \ , \\ (\nu', 1) &= \left(\frac{(k\alpha - \beta)\nu + \alpha\beta(1-k)}{(k-1)\nu + \alpha - k\beta}, 1 \right) \ . \end{aligned} \tag{3}$$

The renormalization coefficient must be strictly positive. Any signal meeting the frontier must cross it; *i.e.* if it comes from below (from above) must go on above (below); this means that $\beta < \nu \Leftrightarrow \beta' < \nu$. To simplify the equality case, we also request that $\nu \neq \beta$. It is now possible to state all the constraints on α, β and k:

$$\left\{ \begin{array}{l} 0 < k, \\ \alpha \neq \beta \ , \\ \\ \forall \nu \in S(M), \left\{ \begin{array}{rcl} \nu & \neq & \beta \ , \\ 0 & < & \lambda = \dfrac{(k-1)\nu + \alpha - k\beta}{\alpha - \beta} \ , \\ \beta < \nu & \Leftrightarrow & \beta < \nu' = \dfrac{(k\alpha - \beta)\nu + \alpha\beta(1-k)}{(k-1)\nu + \alpha - k\beta} \ . \end{array} \right. \end{array} \right. \tag{4}$$

This condition is indeed satisfiable. For example, the conditions are easily verified when

$$0 < k \quad \wedge \quad \alpha < \min(S(M)) \leq \max(S(M)) < \beta \tag{5}$$

since there are only finitely many signals. This represents quite a wide range of values.

4.3 Correctness of the strain

Lemma 7 *Let $\mathcal{M}$ be any signal machine and $\mathcal{M}'$ be the one produced by the algorithm on Fig. 11 for some valid values of α, β, and k. Let c be any (finite) initial configuration for $\mathcal{M}$ and c' be a initial configuration of $\mathcal{M}'$ equals to c except for an additional* toggle *signal on the far left (i.e. it is the leftmost signal). Let us take its position as origin on the spacial axis. Let $\mathbb{D}$ be the space-time diagram issued from c, we suppose that this diagram does not have any accumulation (i.e. no ❋). The space-time diagram issued from c' is as follows:*

Input:

$\mathcal{M}$: signal machine
α: rational { *speed of the strained direction* }
β: rational { *speed of the frontier* }
k: rational { *coefficient of the strain* }

Require:

Equation (4) holds

Do:

{ *Meta-signals addition* }
toggle $\leftarrow \mathcal{M}$.add_meta-signal_of_speed(β)
for all μ meta-signal of $\mathcal{M}$ **do**
  $\mu' \leftarrow \mathcal{M}$.add_meta-signal_of_speed($((k\alpha-\beta)\nu + \alpha\beta(1-k)) \,/\, ((k-1)\nu + \alpha - k\beta)$)
  $\mathcal{M}$.add_rule({ toggle, μ } $\rightarrow$ { toggle, μ' })
  $\mathcal{M}$.add_rule({ toggle, μ' } $\rightarrow$ { toggle, μ })
end for
{ *Rules addition* }
for all rule $\{\mu_i^-\}_i \rightarrow \{\mu_j^+\}_j$ of $\mathcal{M}$ **do**
  $\mathcal{M}$.add_rule($\{\mu_i^{-\prime}\}_i \rightarrow \{\mu_j^{+\prime}\}_j$)
  $\mathcal{M}$.add_rule($\{\mathsf{toggle}\} \cup \{\mu_i^-\}_{\mu_i^-.\nu<\beta} \cup \{\mu_i^{-\prime}\}_{\mu_i^-.\nu>\beta} \rightarrow \{\mathsf{toggle}\} \cup \{\mu_j^+\}_{\mu_j^+.\nu>\beta} \cup \{\mu_j^{+\prime}\}_{\mu_j^+.\nu<\beta}$)
  $\mathcal{M}$.add_rule($\{\mathsf{toggle}\} \cup \{\mu_i^{-\prime}\}_{\mu_i^-.\nu<\beta} \cup \{\mu_i^-\}_{\mu_i^-.\nu>\beta} \rightarrow \{\mathsf{toggle}\} \cup \{\mu_j^{+\prime}\}_{\mu_j^+.\nu>\beta} \cup \{\mu_j^+\}_{\mu_j^+.\nu<\beta}$)
end for

Output:

toggle: toggling meta-signal

Figure 11: Algorithm to build a strain.

1. *the trace of* toggle *is the straight line of equation* $t = \beta x$ *(except for collisions);*
2. *all signals on the left of* toggle *are strained while all the signals on its right are unstrained;*
3. *the part at the right of this line is the same as the space-time diagrams issued from* c*;*
4. *the part at the left of this line is the same as the space-time diagrams issued from* c *strained with coefficients* $(1, k)$ *in the basis* $((1, \beta), (1, \alpha))$.

Before proving this lemma, let us indicate that the space-time diagram on the frontier line ($t = \beta x$) should corresponds to both left and right part, except for the presence of toggle. It should also be noted that the construction on Fig. 11 only add meta-signals and rules; nothing already present is modified, so that as long as no strained signal nor toggle is concern, nothing changes.

Proof The lemma is proved by induction on the dates of collisions. Let us first consider the time between any two collisions, before the first one or after the last one (if any).

Away from collisions, signals undergo uniform movement. Let us suppose that the conditions are verified and prove that they remain satisfied until the next collision. Signals cannot go from one side of toggle to the other without colliding with toggle so that each side can be considered independently. On the right side, unstrained signals follow unstrained dynamic, so that things go as in $\mathbb{D}$. On the left side, the speed of each strained

signal has been calculated so that its trace matches the strained trace of the corresponding unstrained signal.

Let us consider a collision. If it is (strictly) on the right of toggle, then everything happens exactly as in the original case for the collision and outgoing signals. If it is (strictly) on the left of toggle, then everything happens exactly as in the original case, except for the position (by continuity, it is at the strained location), strained version of in-coming signals are replaced by strained versions of out-going signals following the original unstrained rule (each and only such rules are generated on line 13).

Let us consider the last and more complex case: toggle is involved in the collision. If there is only one other signal involved: a (un)strained signal from the left (right). This first sub-case is handled by the rules generates by line 10 (line 9) – again, each and every such rules is generated. The strained signal must come from the left, then its speed satisfies $\beta < \nu'$, then, by condition (4), $\beta < \nu$. This means that the unstrained version of the signal is generated and because of its greater speed it goes on the right of toggle. It passed the frontier rightward and get unstrained. Because the frontier is also the place where the two parts of the diagram agrees, the lemma is verified afterwards. The same holds for the leftward (straining) crossing.

The last sub-case is when more than one signal are involved in a collision with toggle. There are strained in-coming signals on the left (each of their speeds is such that $\beta < \nu'$) and unstrained in-coming signals on the right (each of their speeds is such that $\nu < \beta$). The rule used has been generated by line 14 according to the totally unstrained toggle-less original rule. The outgoing signals are the same as in $\mathbb{D}$ except for the regeneration of toggle and the strain of the signals on its left.

Since there is no accumulation, there is either finitely many collisions or infinitely many but finitely many during any finite duration (otherwise, since all the configurations are inside the light-cone of the initial configuration, there would be infinitely many collisions in a bounded part of the space-time diagram). This means that the induction on the dates of the collisions covers all times.

Q.E.D.

The same holds when toggle is the rightmost signal. If there is an accumulation, little can be said about what happens in the light-cone issued from it: for conservative signal machine, on the one hand, if the accumulation is away from toggle, then since nothing comes out of it in the conservative case, then everything works fine; on the other hand, if it is located on the path of toggle, it erases it and undefined collisions between strained and unstrained signals might happen. Nevertheless, the same holds (and is proved alike) outside of the light-cone issued from an accumulation on toggle.

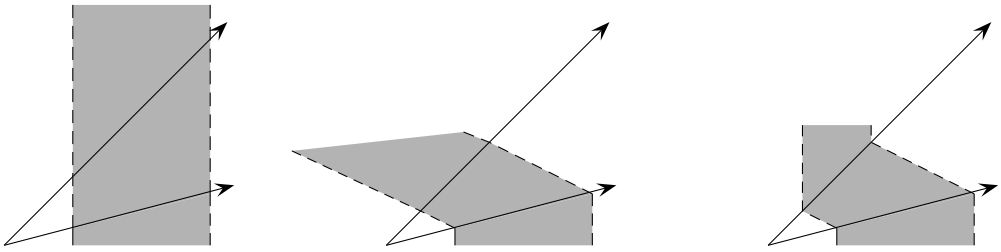


Figure 12: Shrinking principle.

4.4 Shrinking

Theorem 8 *It is possible to scale by* $1/2$ *after some time a space-time diagram.*

Proof The idea is to do two consecutive strains such that afterwards the space-time diagram is scaled by $1/2$, the doubly strained meta-signals are the same as the unstrained ones, and the toggle signals are drifting away from the other signals after the shrinking. This is done very simply: the coefficient is $1/2$ for each strain and the roles of α and β are exchanged so that altogether the two strains combine into a $1/2$ following each axis, thus on any direction and $\nu'' = \nu$ which allows to turn back to the unstrained meta-signals. The critical point is the selection of α and β. We leave to the reader to check that any α and β such that:

$$k = \frac{1}{2} \quad \wedge \quad \max(S(M)) < \beta \quad \wedge \quad 1 < \beta \quad \wedge \quad \alpha = 4\beta \tag{6}$$

satisfy condition (4). Since α and β are both greater than any speed of both strained and unstrained meta-signals (and positive), it is sure that starting on the far left, they will eventually meet all signals and then drift away on the right.

Q.E.D.

Since unstrained (but shrunk) space-time diagram is regenerated, if another pair of toggle signals are added sufficiently far on the left, the space-time diagram is shrunk again, this time to scale $1/4$. In the next subsection, this idea is put into practice with extra signals in charge of generating an endless wave of such pairs of toggle signals.

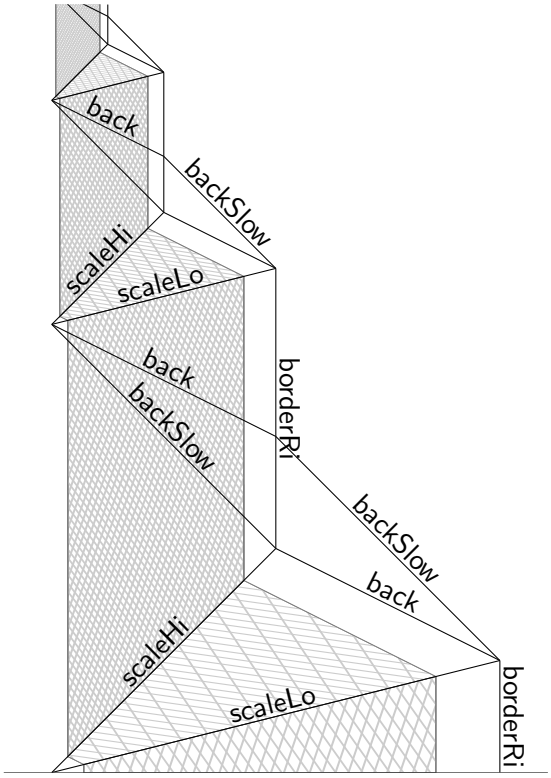
4.5 Iteration

From now on, only spatially bounded space-time diagrams without accumulation are considered.

The idea is to embed a space-time diagram into the shrinking structure on the left part of Fig. 14. The right picture represents the application of this structure to a simulation of a 2-counter automaton. The spacial boundedness of the space-time diagram ensures that the computation remains inside the structure when shrinking is iterated.

The shrinking structure is generated by adding extra signals and rules to restart the shrinking each time as detailed in Fig. 13. The straining frontiers are scaleHi and scaleLo. The speed ν_0 is chosen positive and strictly greater than any speed present in the machine. The signals back and backSlow must cross the space-time diagram without interfering with it; this is done by adding rules as in the line 9 of the algorithm of Fig. 11, it is more simple since straining is not addressed.

In each space-time diagrams of Fig. 14, there is an accumulation point: there are infinitely many collisions accumulating to the upper angle of the triangle. This is a "Zeno effect": finite (continuous) duration but infinitely many (discrete) instants. All collisions are in the light-cone ending there (and there is nothing out of it). This corresponds to the accumulation/black hole of Cond. 4 in Def. 2.



Meta-Signal	Speed
back	$-2\nu_0$
backSlow	$-\nu_0$
borderRi	0
scaleHi	ν_0
scaleLo	$4\nu_0$

$$\{\text{scaleLo}, \text{borderRi}\} \to \{\text{back}, \text{backSlow}\}$$
$$\{\text{scaleHi}, \text{back}\} \to \{\text{backSlow}, \text{borderRi}\}$$
$$\{\text{borderRi}, \text{backSlow}\} \to \{\text{back}, \text{borderRi}\}$$
$$\{\text{backSlow}, \text{back}\} \to \{\text{scaleHi}, \text{scaleLo}\}$$

Figure 13: Structure, meta-signals and rules for the iterated shrinking.

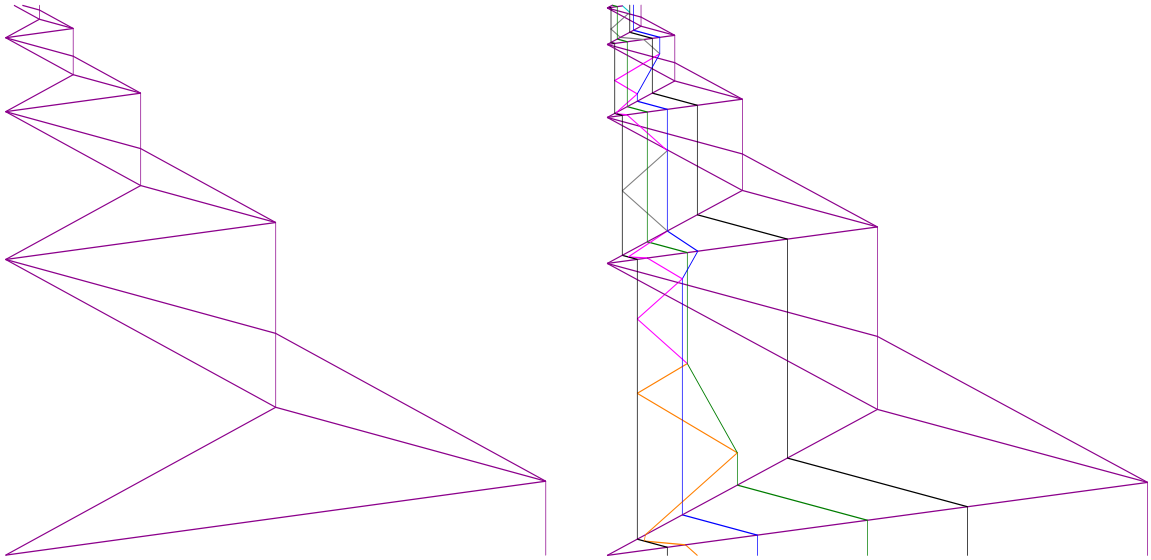


Figure 14: Iterated shrinking: the structure and an example.

5 Black hole formation

We consider the simulation of a 2-counter automata such that, when the simulation stops, if the configuration corresponds to acceptance, then the instruction signal goes on the left. In the case of rejection, another signal would be issued.

The iterated shrinking construction is modified in order not to act on this signal

(*i.e.* it is always generated unstrained and never strained) so that it leaves the iterated shrinking. The iterated shrinking provides the black hole effect; these specially treated signals represent the information that "leaves" the black hole before the collapsing.

It only remains to get this information or assert that no information had left (*i.e.* the computation never stops). This is done by bounding the iterated shrinking by 2 signals that meet after the black hole. If a notification of acceptance or rejection leaves, they grab it before they meet. So that, at the meeting, they know whether the computation finished. This is illustrated by Fig. 15.

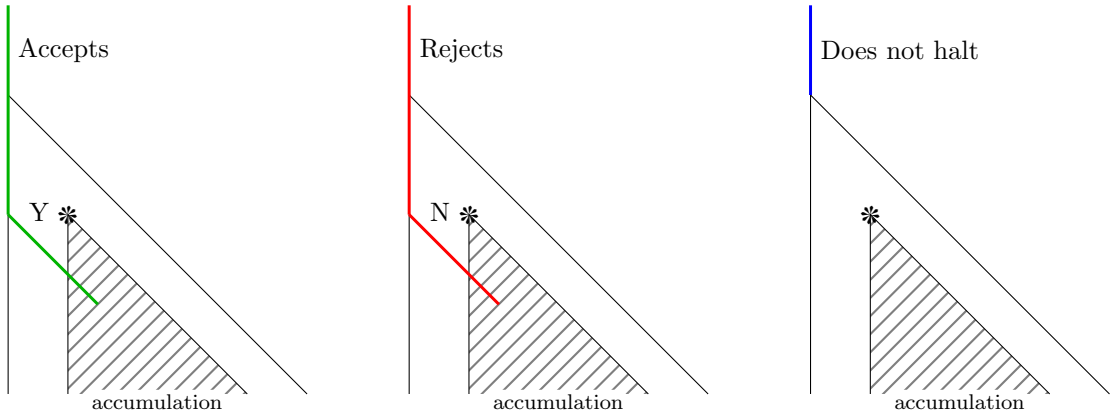


Figure 15: Encapsulation of a black hole.

6 Conclusion

We provide a geometrical model of computation that is Turing-computation universal and has the special features of the Black hole model, in particular it can decide any $\mathcal{R}.\mathcal{E}.$ problem (more on abstract geometrical computation can be found in [DL03]). We are not using already existing black holes, but rather creating them on demand (a Malament-Hogarth space-time is implicitly built). It is not so strange that computation forms the black hole since, in the black hole setting, they "come from" the "same matter" as the machine sent into it. One can also consider that some signals fix black hole formation, while others carry out the computation using the black hole.

One may object that black holes disappearance is not acceptable. The underlying space being one-dimensional, any remaining black hole would form a barrier preventing information to cross from one side to the other; whereas in dimension two and more, it is always possible to go around it. Another argument is to imagine that signals are drifting in a higher dimensional space, so that the black holes remain, but its orbit is not in the plane of the space-time diagram.

Reversibility is an important issue in theoretical physics. One can easily check that reversibility corresponds to R being one-to-one. At the expense of more involved constructions, universality can be achieved as well as the use of accumulations as black holes. But the final collapsing is not reversible.

The number of possible black holes/$\mathcal{R}.\mathcal{E}.$ queries is bounded from the start (each one needs a minimal amount of energy). Unless the black hole returns the energy in some form,

which is clearly forbidden here, there is no way to address second order accumulations (*i.e.* ω^2 or second order space-time arithmetic deciding or Σ_2^0 in the arithmetical hierarchy), unless infinitely many signals exist at start, apart one from another. This way it would be reasonably possible to hope to climb the arithmetical hierarchy as in [AM95, Bou99].

As long as the model is restricted to rational numbers, there are finitely many signals present at any instant and there is no accumulation, the model is Turing-universal and can be simulated by any Turing machine and is thus Turing-equivalent. If the "finite number of signals" condition is lifted, but signals are isolated, then this is a super-Turing model of computation following the "Infinity principle" of [EW03]. The analog power really appears when the rationality constraint is lifted. Real values for speeds and/or positions can be used as oracles and thus provide computing ability that goes beyond classical Turing-computation.

References

[Ada02] A. Adamatzky, editor. *Collision based computing*. Springer, 2002.

[AM95] E. Asarin and O. Maler. Achilles and the Tortoise climbing up the arithmetical hierarchy. In *FSTTCS '95*, number 1026 in LNCS, pp. 471–483, 1995.

[BNR91] N. Boccara, J. Nasser, and M. Roger. Particle-like structures and interactions in spatio-temporal patterns generated by one-dimensional deterministic cellular automaton rules. *Phys. Rev. A*, 44(2):866–875, 1991.

[Bou99] O. Bournez. Achilles and the Tortoise climbing up the hyper-arithmetical hierarchy. *Theoret. Comp. Sci.*, 210(1):21–71, 1999.

[DL97] J. Durand-Lose. Intrinsic universality of a 1-dimensional reversible cellular automaton. In *STACS '97*, number 1200 in LNCS, pp. 439–450. Springer, 1997.

[DL03] J. Durand-Lose. *Calculer géométriquement sur le plan – machines à signaux*. Habilitation à diriger des recherches, École Doctorale STIC, Université de Nice-Sophia Antipolis, 2003.

[DL05] J. Durand-Lose. Abstract geometrical computation for black hole computation (extended abstract). In M. Margenstern, editor, *Universal Machines and Computations (MCU '04)*, number 3354 in LNCS, pp. 175–186. Springer, 2005.

[DM02] M. Delorme and J. Mazoyer. Signals on cellular automata. in [Ada02], pp. 234–275, 2002.

[EN02] G. Etesi and I. Nemeti. Non-Turing computations via Malament-Hogarth space-times. *Int. J. Theor. Phys.*, 41(2):341–370, 2002. gr-qc/0104023.

[EW03] E. Eberbach and P. Wegner. Beyond Turing machines. *Bull. EATCS*, 81:279–304, 2003.

[Fis65] P. C. Fischer. Generation of primes by a one-dimensional real-time iterative array. *J. ACM*, 12(3):388–394, 1965.

[Got66] E. Goto. Ōtomaton ni kansuru pazuru [Puzzles on automata]. In T. Kitagawa, editor, *Jōhōkagaku eno michi [The Road to information science]*, pp. 67–92. Kyoristu Shuppan Publishing Co., Tokyo, 1966.

[Ham02] J. D. Hamkins. Infinite time Turing machines: Supertask computation. *Minds and Machines*, 12(4):521–539, 2002. arXiv:math.LO/0212047.

[HL00] J. D. Hamkins and A. Lewis. Infinite time turing machines. *J. Symb. Log.*, 65(2):567–604, 2000. arXiv:math.LO/9808093.

[Hog94] M. Hogarth. Non-Turing computers and non-Turing computability. In *Biennial Meeting of the Philosophy of Science Association*, number 1, pp. 126–138, 1994.

[Hog00] M. Hogarth. *Predictability, computability and space-time*. PhD thesis, University of Cambridge, UK, 2000. ftp://ftp.math-inst.hu/pub/algebraic-logic/Hogarththesis.ps.gz.

[HSC01] W. Hordijk, C. R. Shalizi, and J. P. Crutchfield. An upper bound on the products of particle interactions in cellular automata. *Phys. D*, 154:240–258, 2001.

[Ila01] A. Ilachinski. *Cellular automata –a discrete universe*. World Scientific, 2001.

[JS90] G. Jacopini and G. Sontacchi. Reversible parallel computation: an evolving space-model. *Theoret. Comp. Sci.*, 73(1):1–46, 1990.

[JSS02] M. H. Jakubowsky, K. Steiglitz, and R. Squier. Computing with solitons: a review and prospectus. in [Ada02], pp. 277–297, 2002.

[LN90] K. Lindgren and M. G. Nordahl. Universal computation in simple one-dimensional cellular automata. *Complex Systems*, 4:299–318, 1990.

[LN04] S. Lloyd and Y. J. Ng. Black hole computers. *Scientific American*, 291(5):31–39, November 2004.

[Maz96a] J. Mazoyer. Computations on one dimensional cellular automata. *Annals of Mathematics and Artificial Intelligence*, 16:285–309, 1996.

[Maz96b] J. Mazoyer. On optimal solutions to the Firing squad synchronization problem. *Theoret. Comp. Sci.*, 168(2):367–404, 1996.

[Min67] M. Minsky. *Finite and infinite machines*. Prentice Hall, 1967.

[MT99] J. Mazoyer and V. Terrier. Signals in one-dimensional cellular automata. *Theoret. Comp. Sci.*, 217(1):53–80, 1999.

[Siw01] P. Siwak. Soliton-like dynamics of filtrons of cycle automata. *Inverse Problems*, 17:897–918, 2001.

[VMP70] V. I. Varshavsky, V. B. Marakhovsky, and V. A. Peschansky. Synchronization of interacting automata. *Math. System Theory*, 4(3):212–230, 1970.