



**HAL**  
open science

## Drawing numerable linear orderings

Jérôme Durand-Lose Laboratoire d'Informatique Fondamentale d'Orleans

► **To cite this version:**

Jérôme Durand-Lose Laboratoire d'Informatique Fondamentale d'Orleans. Drawing numerable linear orderings. *Unconventional Computation and Natural Computation*, 2023, Jacksonville - Floride, United States. 2023. hal-04355447

**HAL Id: hal-04355447**

**<https://hal.science/hal-04355447>**

Submitted on 20 Dec 2023

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



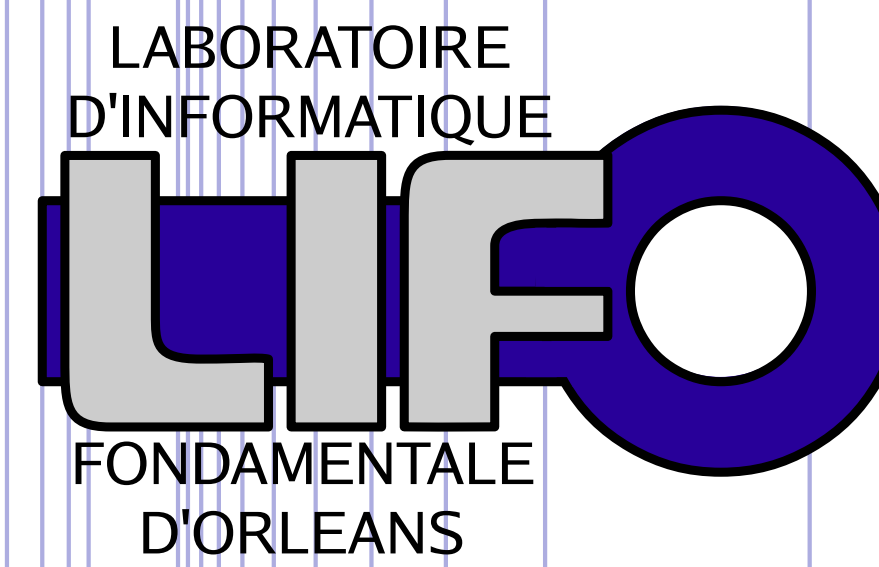
Distributed under a Creative Commons Attribution - NonCommercial - ShareAlike 4.0 International License





# Drawing numerable linear orderings

Jérôme DURAND-LOSE



Laboratoire d'Informatique Fondamentale d'Orléans  
 ÉA 4022  
 Université d'Orléans, ORLÉANS, FRANCE  
 Funded by ANR DIFFERENCE (ANR-20-CE40-0002)



## Numerable linear orderings ( $S, \leq_S$ )

- $S$  is a numerable set
- $\leq_S$  is a reflexive, anti-symmetric and transitive relation on  $S$
- any two elements of  $S$  are related/comparable
- examples:
  - $\omega$  : the ordering of the natural numbers  $(0, 1, 2, \dots)$
  - $\zeta$  : the ordering of the integers  $(\dots, -2, -1, 0, 1, 2, \dots)$
- operations
  - addition: set union and all the elements from a set before the elements from the other
  - product: set product and with lexicographical order
  - $\{a, b\}^*$  with lexical order
- examples:
  - $\omega + \omega$  is  $0, 1, 2, \dots, 0', 1', 2', \dots$  ( $0'$  is a distinct copy of 0)
  - $\omega \cdot \omega$  is  $(0, 0), (0, 1), (0, 2), \dots, (1, 0), (1, 1), (1, 2), \dots, (2, 0), (2, 1), (2, 2), \dots$

## Ordinals

- well founded orderings
- examples:
  - $\omega, \omega + \omega, \omega + \omega + \omega, \omega \cdot \omega, \omega \cdot \omega \cdot \omega + \omega \cdot \omega + \omega$
- non-ordinal linear orderings:
  - $\zeta$  has the infinite decreasing sequence  $(\dots, -2, -1, 0)$
  - $\{a, b\}^*$  with lexical order has the infinite decreasing sequence  $(\dots, aab, ab, b)$

- More on linear orderings  $\rightsquigarrow$  Rosenstein [1982]

## Decidable

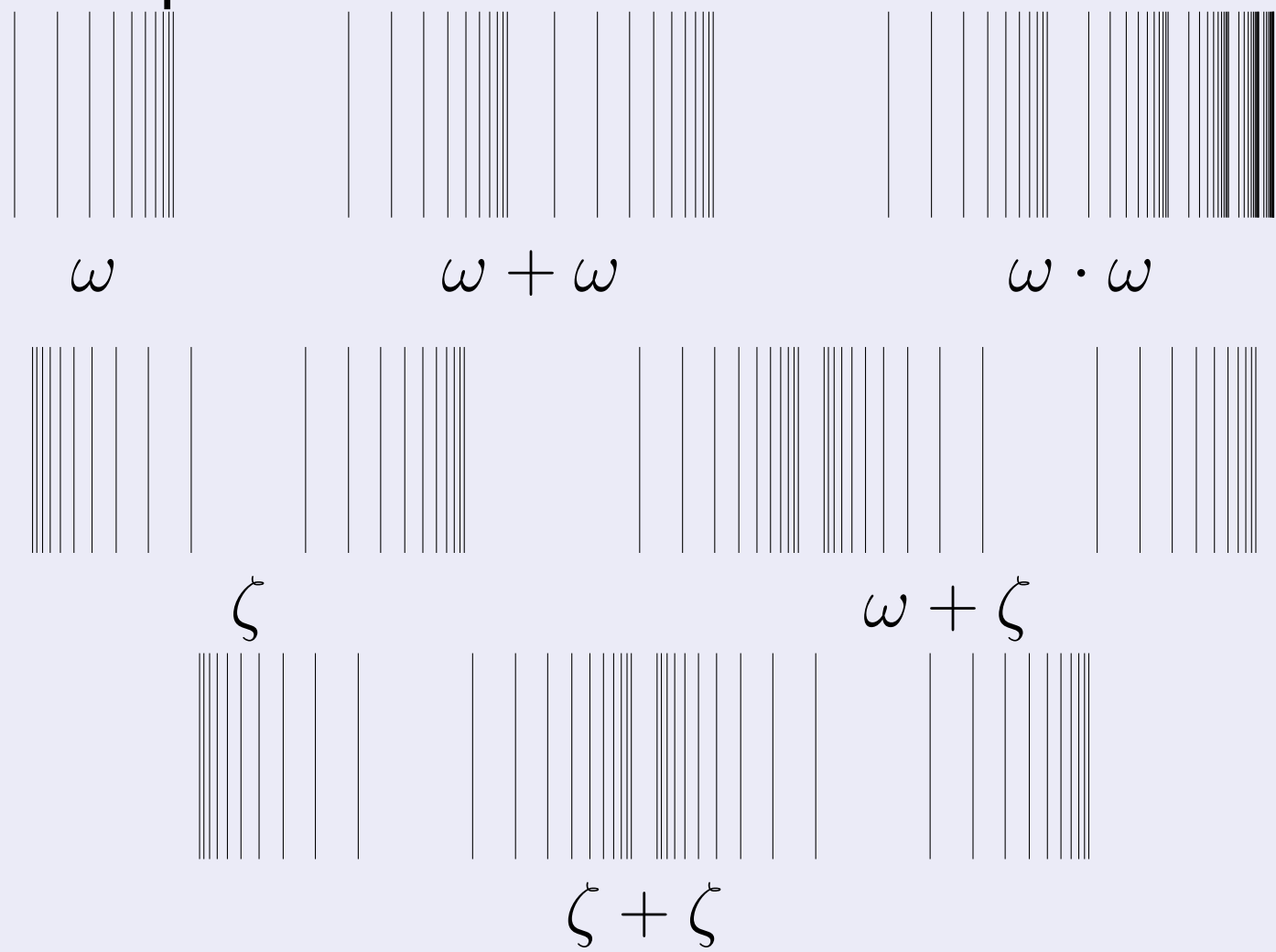
- $s_0, s_1, s_2, \dots$  : an enumeration of the elements of  $S$
- A Turing machine decides the relation

$$i, j \mapsto s_i \leq_S s_j$$

## Graphical representation

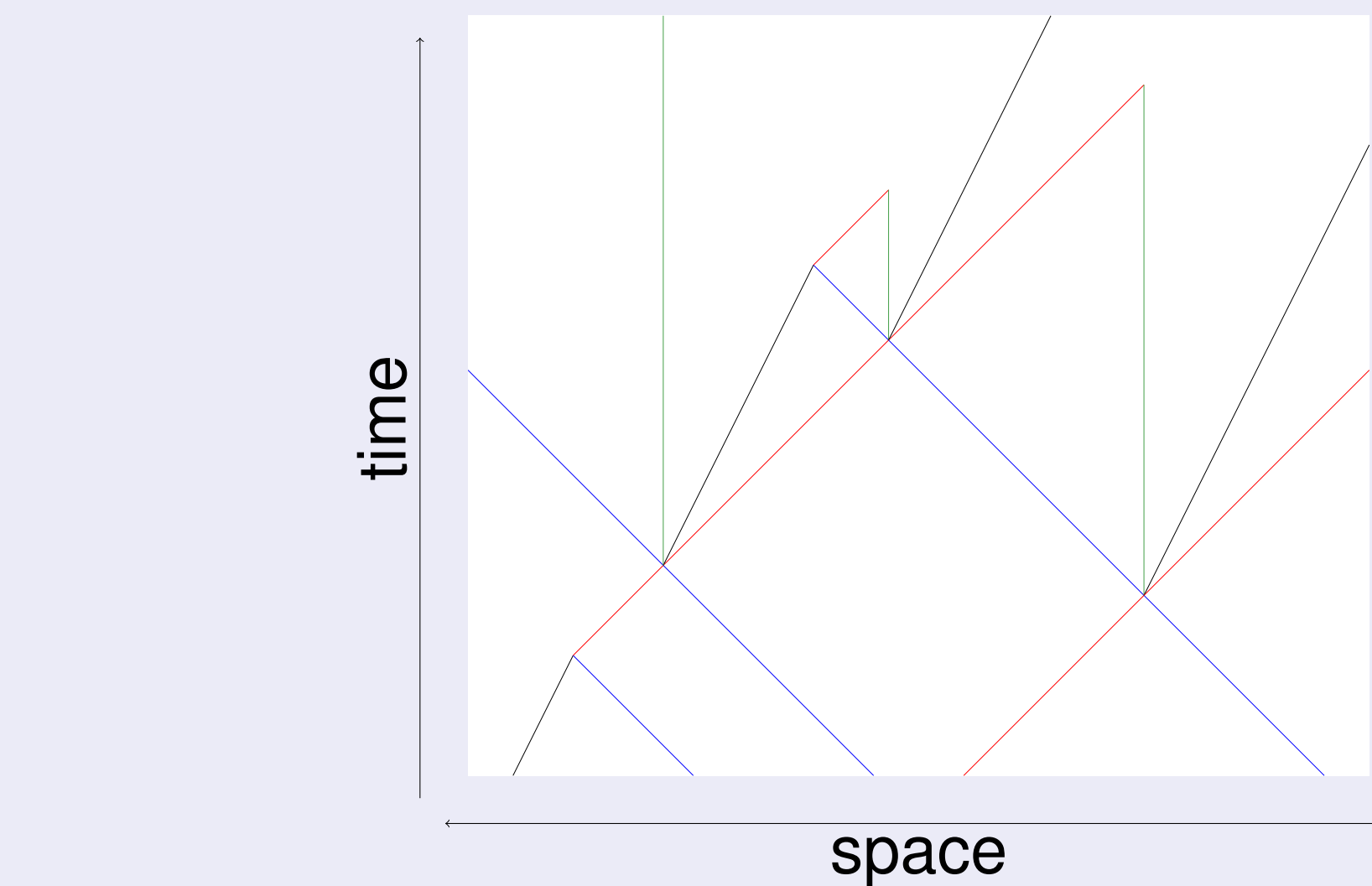
- by parallel vertical lines s.t.:
  - one line per element in  $S$
  - if  $x \leq_S y$  then  $x$  is on the left of  $y$
  - there is an empty space between  $x$  and  $y$  ( $x \leq_S y$ ) iff  $x$  is the immediate predecessor of  $y$
  - iff  $y$  is the immediate successor of  $x$
  - iff  $\forall z, x \leq_S z \leq_S y \rightarrow x = z = y$

- examples:

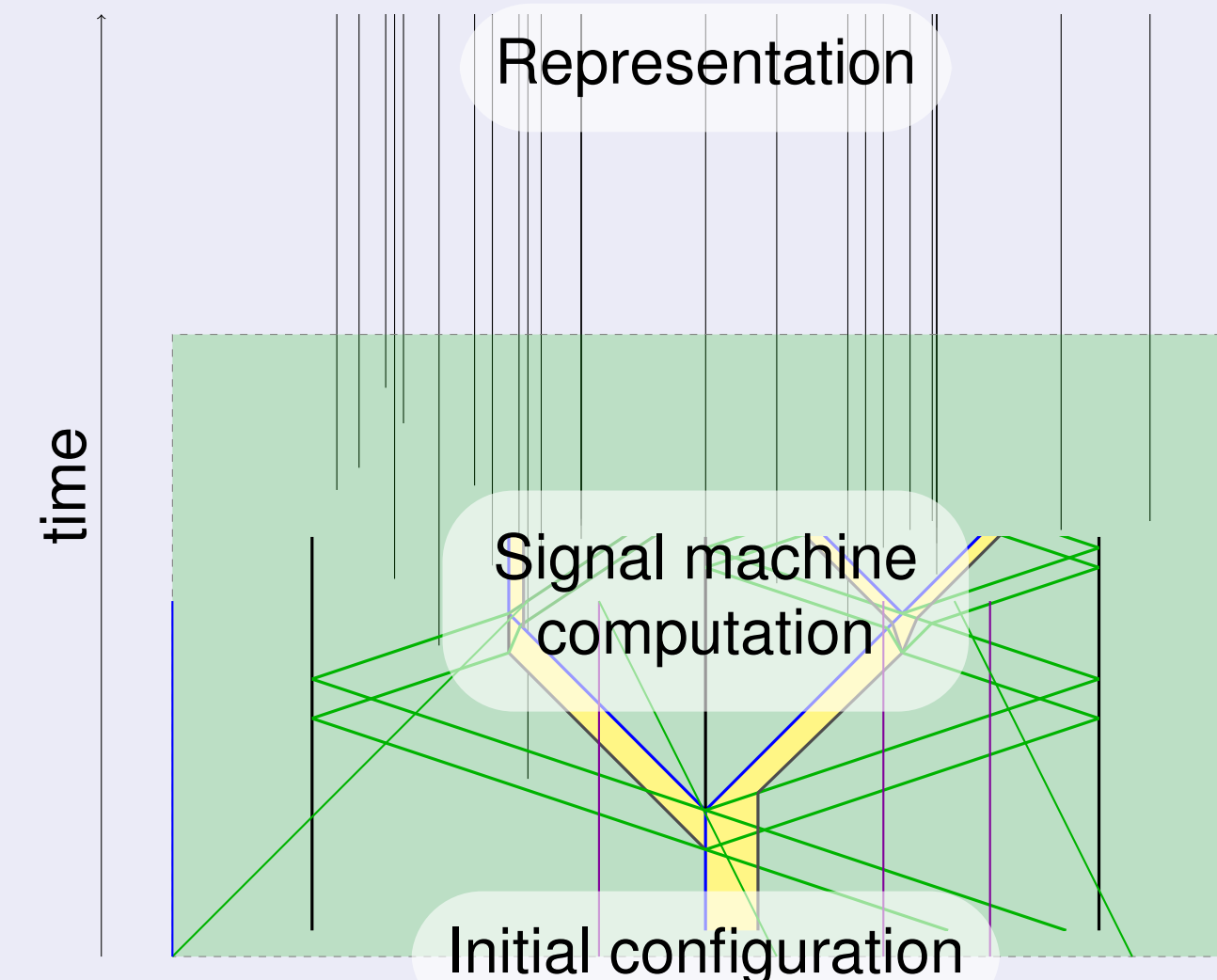


## Signal Machines

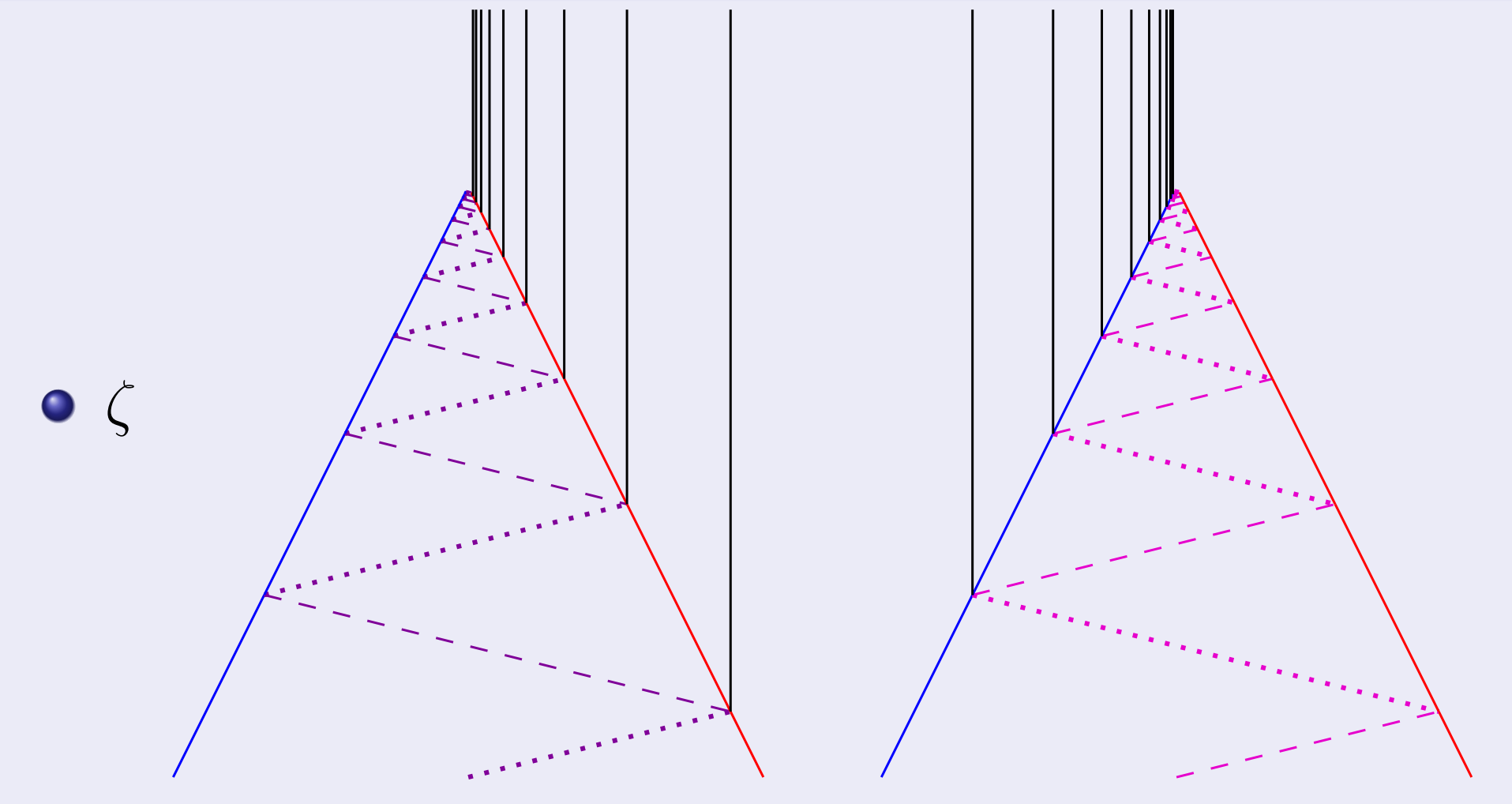
- signals
  - red, black, green, blue constant speed for each kind
- collision rules
  - $\{\text{black}, \text{blue}\} \rightarrow \{\text{red}\}$
  - $\{\text{red}, \text{blue}\} \rightarrow \{\text{blue}, \text{green}, \text{black}, \text{red}\}$
- dynamics  $\rightsquigarrow$  space-time diagrams



## Global scheme



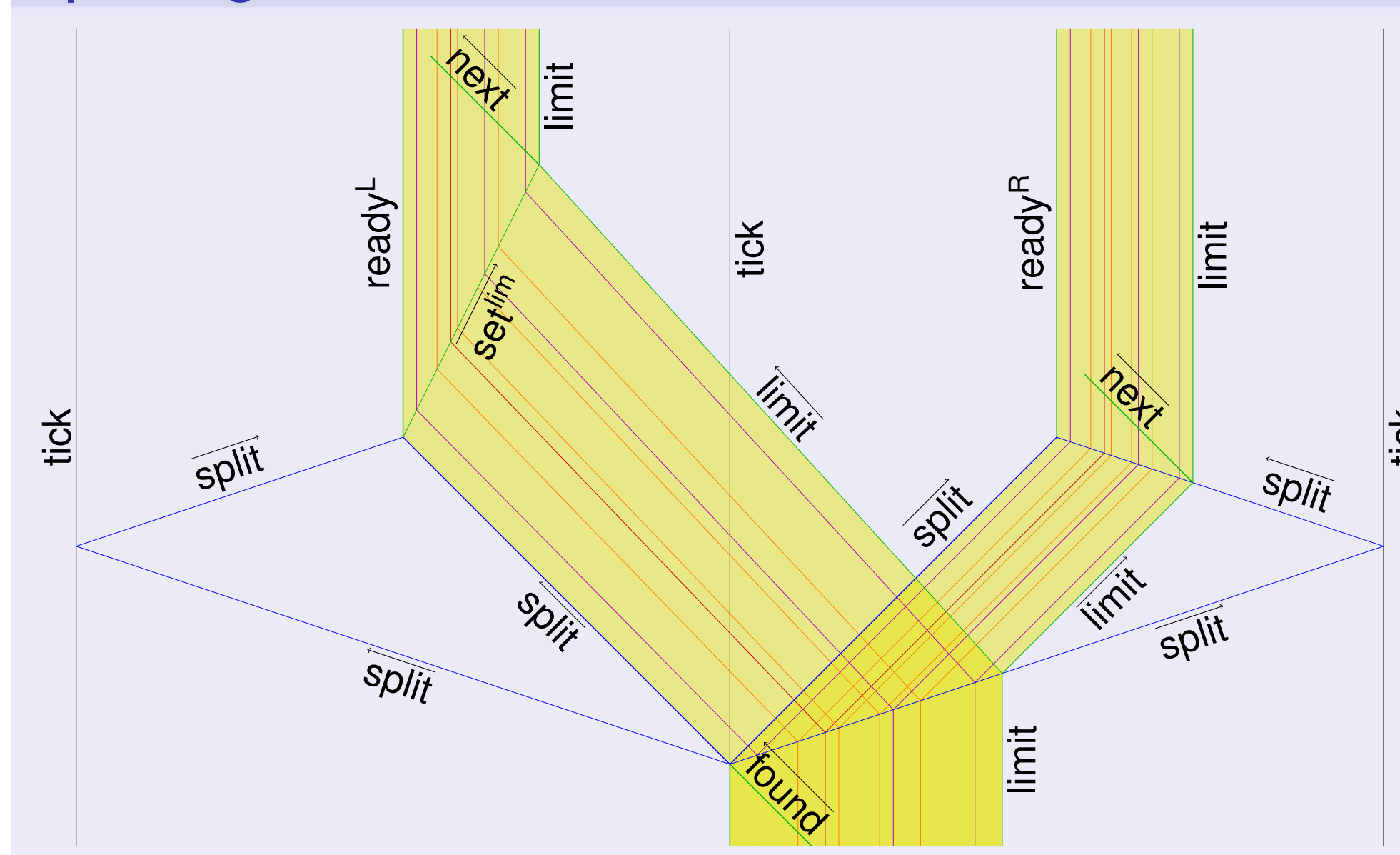
## Ad hoc constructions



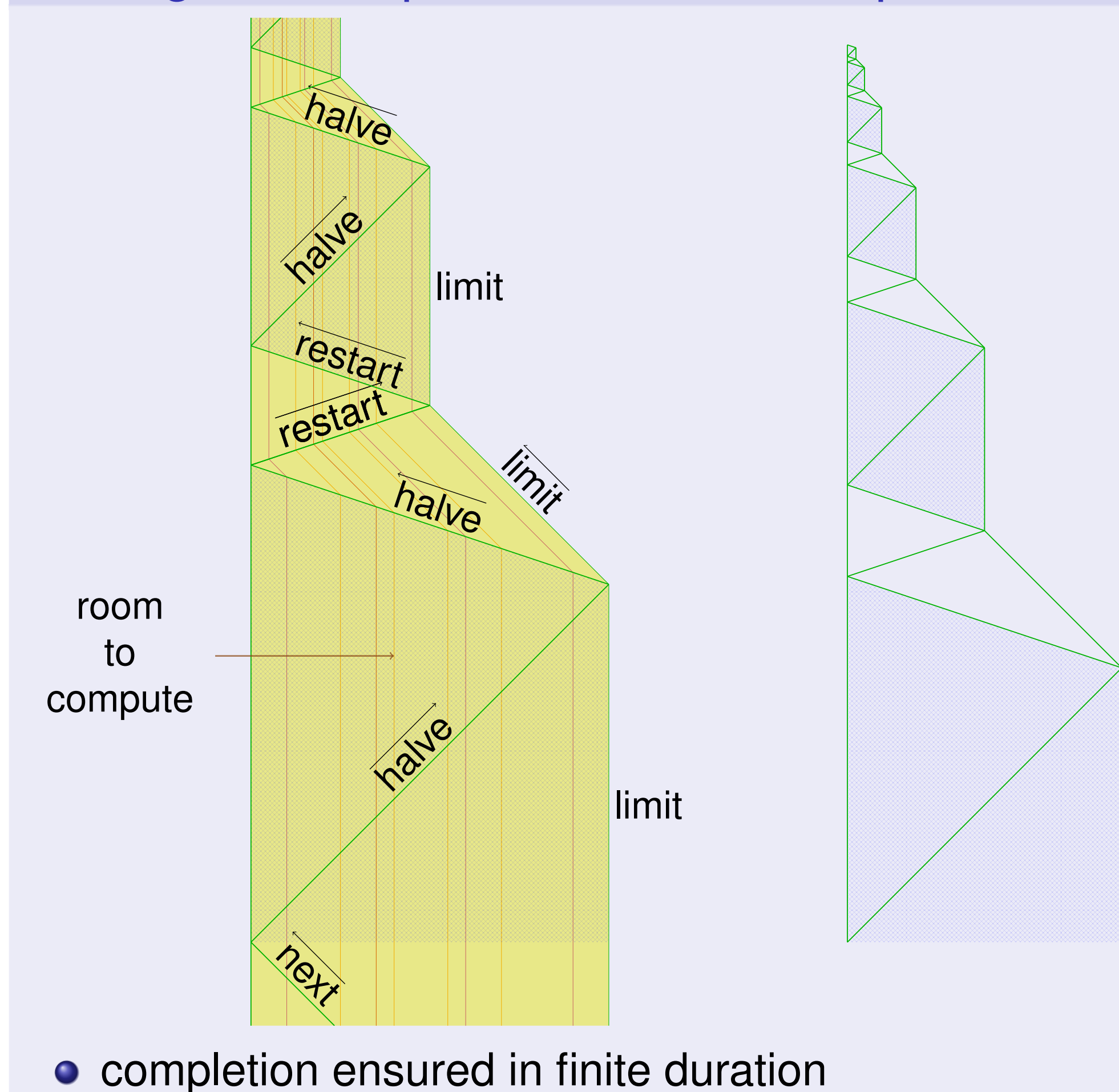
## Algorithm

- let  $i, j$  be two indices s.t. lines are set
- find least  $k$  such that:
 
$$k \neq i \wedge k \neq j \wedge s_i \leq_S s_k \leq_S s_j$$
- if  $k$  is found, the interval is split and the algorithm is restarted on  $i$  and  $k$  on the left and  $k$  and  $j$  on the right
- if no such  $k$  exists, the computation vanishes
- this is done in finite time

## Splitting

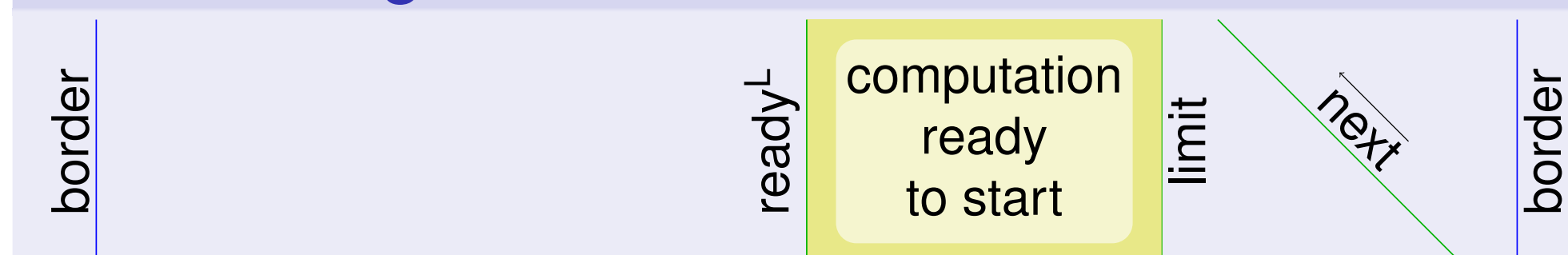


## Scaling the computation at each step



- completion ensured in finite duration

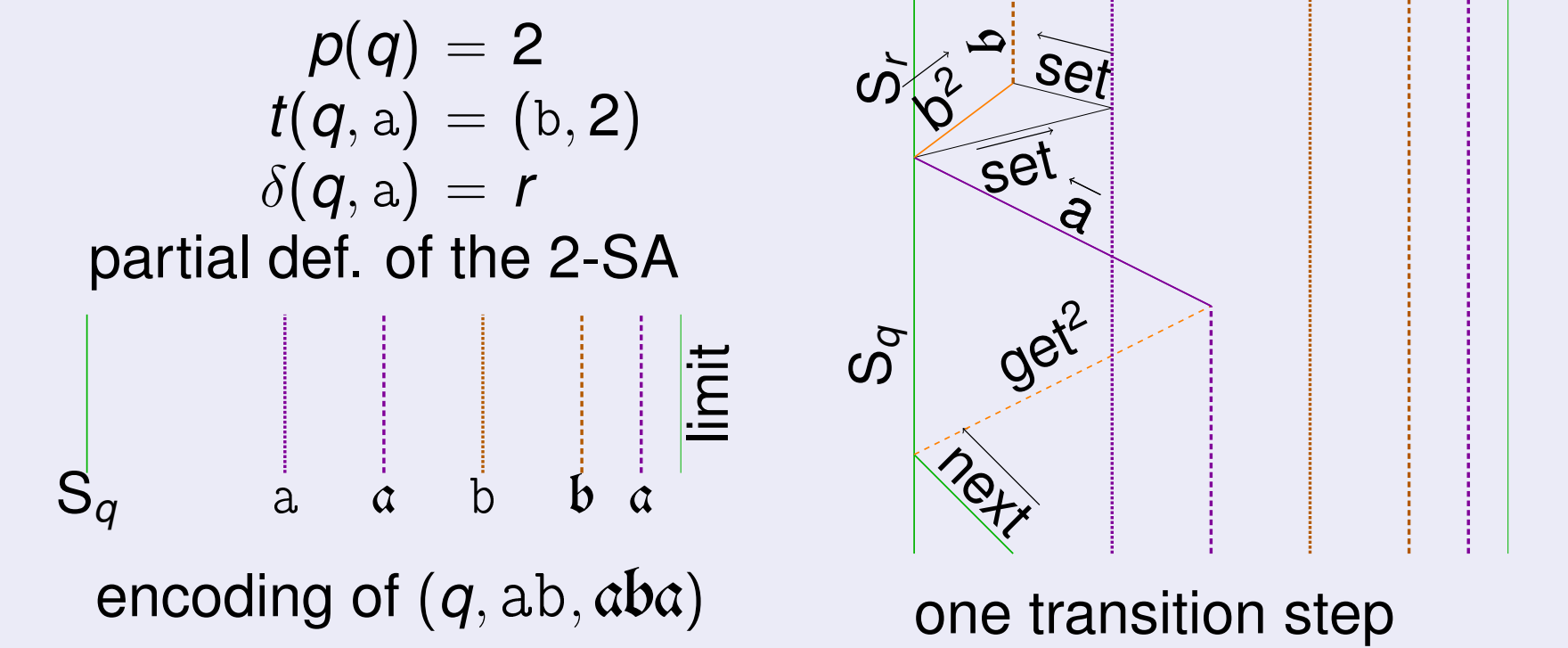
## Initial configuration



## Working on $\mathbb{Q}$

There is a rational signal machine able to generate the representation of any decidable countable linear ordering.

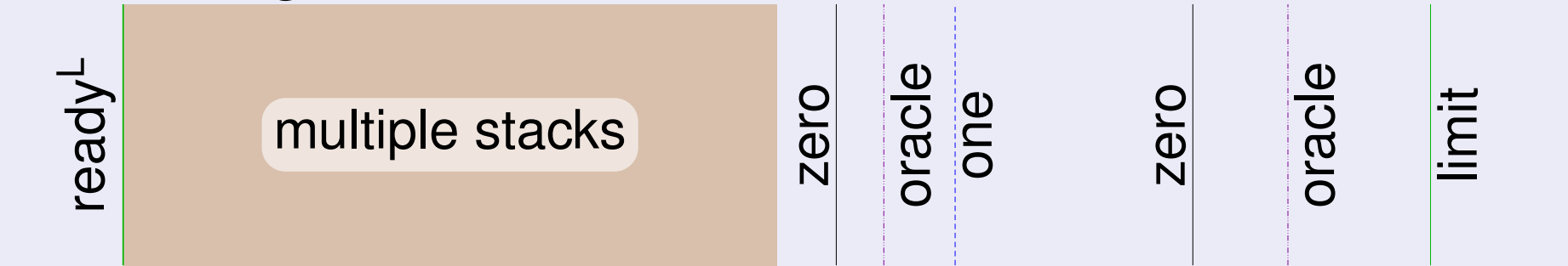
- Simulating multi-stack automaton



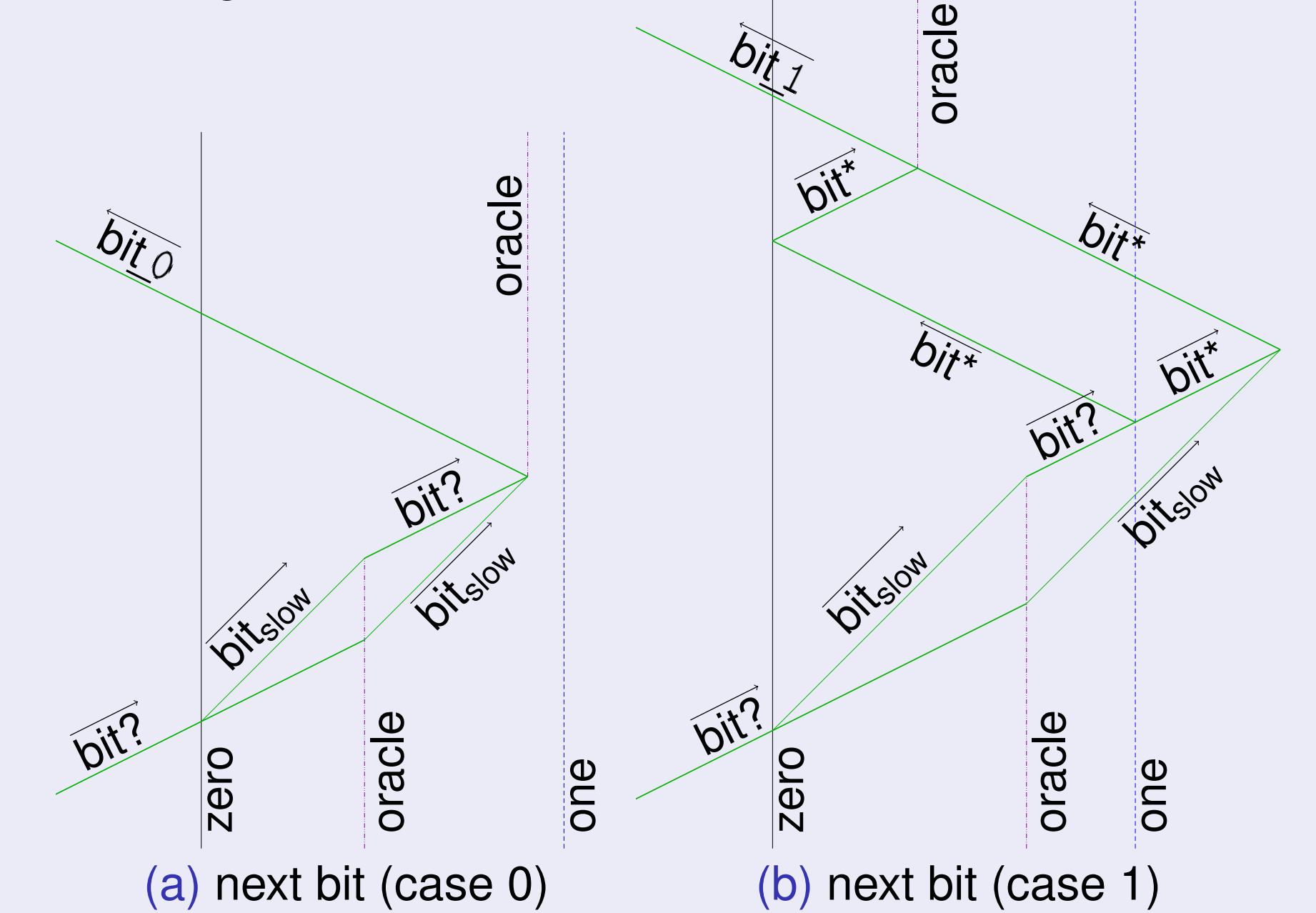
## Working on $\mathbb{R}$

There is a signal machine able to generate the representation of any countable linear ordering.

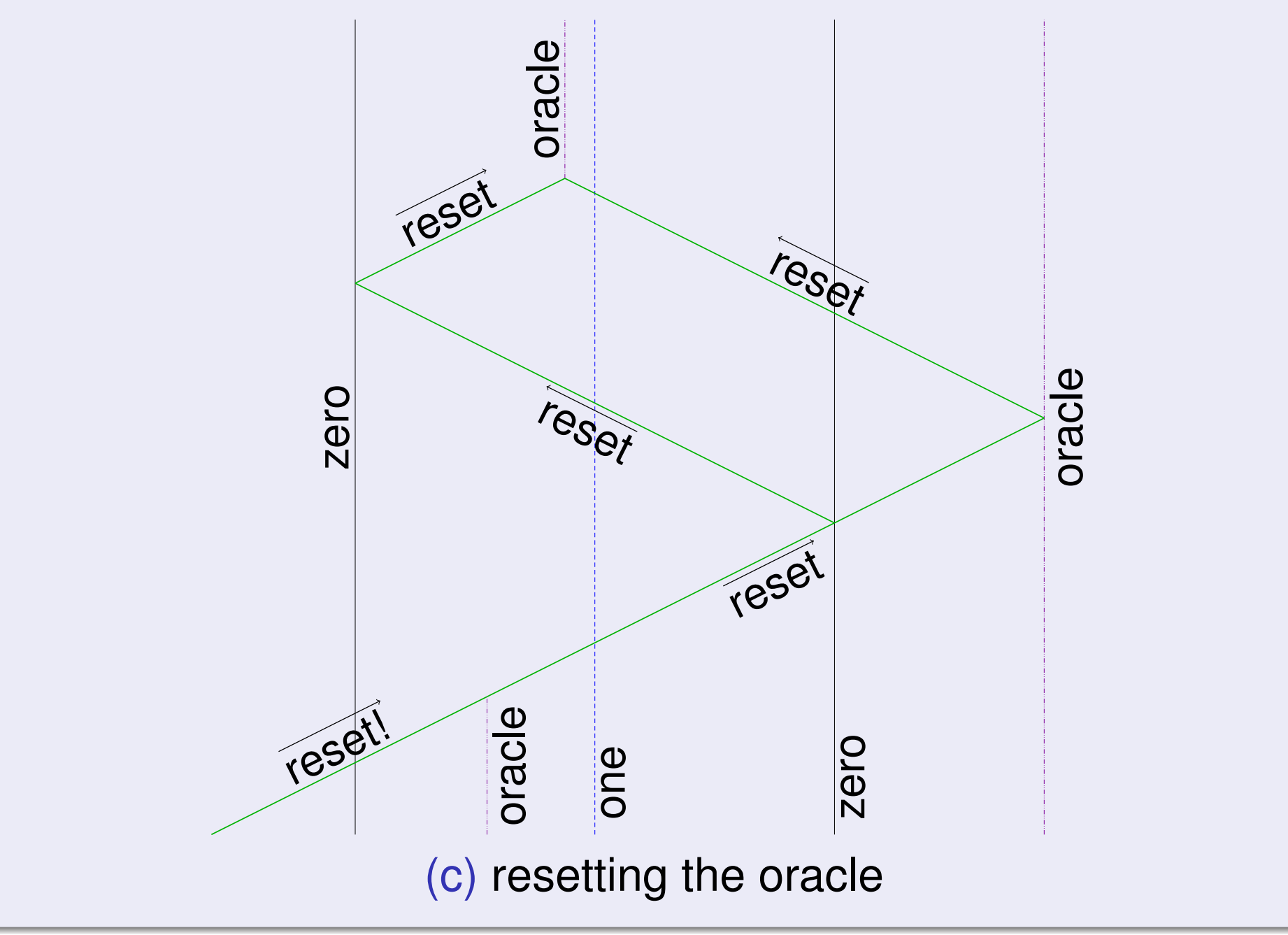
- Encoding an oracle



- Handling the oracle



(a) next bit (case 0) (b) next bit (case 1)



(c) resetting the oracle

## References

J. Durand-Lose. Ways to Compute in Euclidean Frameworks. In M. J. Patitz and M. Stannett, editors, *Unconventional Computation and Natural Computation - 16th Int. Conf., UCNC 2017, Fayetteville, AR*, volume 10240 of LNCS, pages 8–25. Springer, 2017. doi: 10.1007/978-3-319-58187-3\_2.

J. G. Rosenstein. *Linear ordering*. Academic Press, 1982.