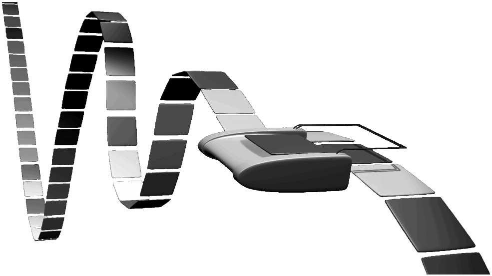
On Aperiodic Reversible Turing Machines

Julien Cassaigne (CNRS, Marseille, France)
Anahí Gajardo (Univ. de Concepción, Chile)
Jarkko Kari (Univ. of Turku, Turku, Finland)
Nicolas Ollinger (LIFO, Univ. Orléans, France)
Rodrigo Torres Avilés (Univ. del Bío-Bío, Chillán, Chile)

RC 2018, Leicester — September 14th, 2018

Turing machines

The classical **Turing machine**: finitely many **states**, a (bi-)infinite **tape**, a mobile i/o **head** pointing on a cell (optionally: blank symbol, starting and halting states).



Halting Problem[Σ^0_1-comp.] Given a TM and a finite starting configuration, decide if a halting state is eventually reached.

Reachability and similar questions

Reachability Problem[Σ^0_1-comp.] Given a TM and two states s and t, decide if state t is reachable from state s.

Totality Problem[Π^0_2-comp.] Given a TM, decide if it eventually halts starting from any **finite configuration**.

Mortality Problem[Σ^0_1-comp.] Given a TM, decide if it eventually halts starting from any configuration.

Periodicity Problem[Σ^0_1-comp.] Given a TM, decide if every configuration eventually loops by reaching itself again.

The Transitivity Problem

Transitivity Problem[Π^0_1-hard] Given a TM, decide if every **partial configuration** is reachable from every **partial configuration** (completed into a proper configuration).

A **partial configuration** specifies only a finite segment of the tape plus the state and head position.

$$???????ab.babaa???????$$
$$\mathbf{q}$$

Question How do we prove the undecidability of the Transitivity Problem?

The Transitivity Problem

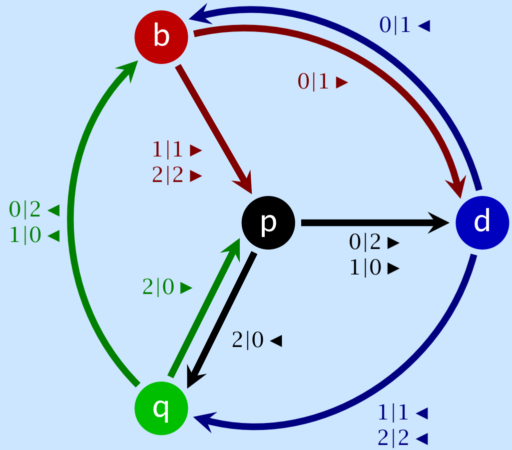
Transitivity Problem[Π^0_1-hard] Given a TM, decide if every **partial configuration** is reachable from every **partial configuration** (completed into a proper configuration).

A **partial configuration** specifies only a finite segment of the tape plus the state and head position.

$$???????ab.ba\underset{\mathbf{q}}{b}aa???????$$

Question How do we prove the undecidability of the Transitivity Problem?

Question ...and first, how do you build a transitive TM?



1. Dynamics of Turing machines

Turing machines

A **Turing machine** is a triple (Q, Σ, δ) where Q is the finite set of states, Σ is the finite set of tape symbols and $\delta : Q \times \Sigma \to Q \times \Sigma \times \{-1, 0, 1\}$ is the partial transition function.

A transition $\delta(s, a) = (t, b, d)$ means:
"in state s, when reading the symbol a on the tape,
replace it by b move the head in direction d and enter state t."

Configurations are triples $(s, c, p) \in Q \times \Sigma^{\mathbb{Z}} \times \mathbb{Z}$.

A **transition** transforms (s, c, p) into $(t, c', p + d)$ where $\delta(s, c(p)) = (t, b, d)$ and $c' = c$ everywhere but $c'(p) = b$.

Notation $(s, c, p) \vdash (t, c', p + d)$ and closures $\vdash^+$ and $\vdash^*$

Definitions

A configuration (s, c, p) is:

- **halting** if $\delta(s, c(p))$ is undefined, $(s, c(p))$ is a **halting pair**
- **periodic** if $(s, c, p) \vdash^+ (s, c, p)$

A TM (Q, Σ, δ) is:

- **complete** if δ is complete
- **aperiodic** if it has no periodic configuration
- **surjective** if every configuration has a preimage
- **injective** if every configuration has at most one preimage

For complete machines surjective is equivalent to injective.

Reversibility

Injective TM are in fact reversible TM.

Definition A **reversible** TM $M = (Q, \Sigma, \delta)$ is characterized by a partial injective map ρ and a map μ such that $\delta(s,a) = (t, b, \mu(t))$ where $\rho(s,a) = (t,b)$.

The **reverse** of M is M^{-1} where $\delta^{-1}(t,b) = (s, a, -\mu(s))$.

$$(s,c,p) \vdash_M (t,c',p+\mu(t)) \implies (t,c',p) \vdash_{M^{-1}} (s,c,p-\mu(s))$$

A **starting pair** is a halting pair of the reverse.

A **starting configuration** is a halting config of the reverse.

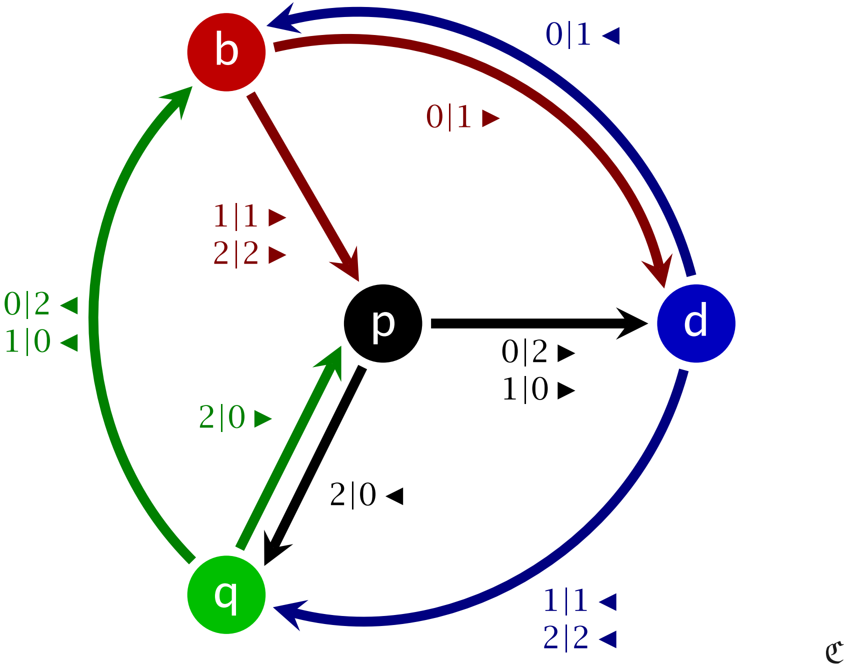
Naive dynamics

A **topological dynamical system** is a pair (X, T) where the topological space X is the **phase space** and the continuous function $T : X \to X$ is the **global transition function**.

The **orbit** of $x \in X$ is $\mathcal{O}(x) = (T^n(x))_{n\in\mathbb{N}}$.

Using the **product topology** one obtains a **topological dynamical system** (X, T) for a TM where the phase space is $X = Q \times \Sigma^{\mathbb{Z}} \times \mathbb{Z}$ and the transition function T is continuous.

Unfortunately, X is not **compact**, we follow Kůrka's alternative compact dynamical models TMH and TMT.



b
p
d
q
0|1 ◀
0|1 ▶
1|1 ▶
2|2 ▶
0|2 ◀
1|0 ◀
0|2 ▶
1|0 ▶
2|0 ▶
2|0 ◀
1|1 ◀
2|2 ◀

Moving head vs moving tape dynamics

TMH

$$X_h \subset (Q \cup \Sigma)^{\mathbb{Z}}$$
$$T_h : X_h \to X_h$$

···000000**b**000000000···
···0000001**d**00000000···
···000000**b**110000000···
···0000001**p**10000000···
···00000010**d**0000000···
···0000001**b**01000000···
···00000011**d**1000000···
···0000001**q**11000000···
···000000**b**101000000···
···0000001**p**01000000···
⋮

TMT

$$X_t = {}^{\omega}\Sigma \times Q \times \Sigma^{\omega}$$
$$T_t : X_t \to X_t$$

···0000000**b**00000000···
···0000001**d**00000000···
···0000000**b**11000000···
···0000001**p**10000000···
···0000010**d**00000000···
···0000001**b**01000000···
···0000011**d**10000000···
···0000001**q**11000000···
···0000000**b**10100000···
···0000001**p**01000000···
⋮

Trace-shift dynamics

ST

$$S_T \subseteq (Q \times \Sigma)^\omega$$

$$\sigma : S_T \to S_T$$

TMT

$$X_t = {}^\omega\Sigma \times Q \times \Sigma^\omega$$

$$T_t : X_t \to X_t$$

The column shift of TMT

0 0 1 1 0 0 1 1 1 0 . . .
b d b p d b d q b p

```
···0000000b00000000···
···0000001d00000000···
···0000000b11000000···
···0000001p10000000···
···0000010d00000000···
···0000001b01000000···
···0000011d10000000···
···0000001q11000000···
···0000000b10100000···
···0000001p01000000···
⋮
```

Topological transitivity

Definition A dynamical system (X,T) is **transitive** is it admits a **transitive point** x such that $\overline{\mathcal{O}(x)} = X$.

Proposition (X,T) is **transitive** iff for every pair of open sets $U,V \subseteq X$, there exists t such that $T^t(U) \cap V \neq \emptyset$.

TMH $\forall u,v,u',v' \; \exists w,z,w',z',n \;\; T_h^n(wu.vz) = w'u'.v'z'$

TMT $\forall u,v,\alpha,u',v',\beta \; \exists w,z,w',z',n \; T_t^n(wu,\alpha,vz) = (w'u',\beta,v'z')$

ST $\forall u,v{\in}S_T \;\; \exists w{\in}S_T \quad uwv \in S_T$

TMH transitive $\Longrightarrow$ TMT transitive $\Longrightarrow$ ST transitive.

TMH transitive implies **complete**, **reversible** and **aperiodic**

Transitivities

Definition A point $x \in X$ is **periodic** if it admits a **period** $p > 0$ such that $T^p(x) = x$.

Proposition A TM with a **periodic** point is **not ST transitive**.

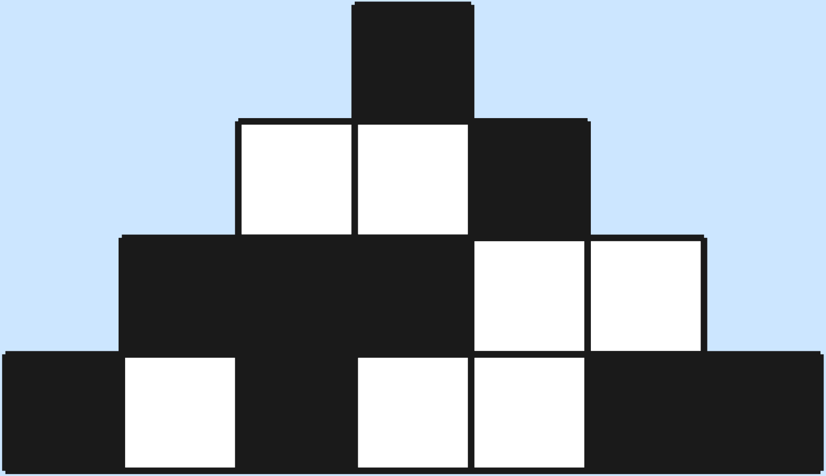
The single-state **shift** TM is **TMT transitive** but **not TMH**.

$$\delta(q, x) = (q, x, \blacktriangleright)$$

The single-state **eraser** TM is **ST transitive** but **not TMT**.

$$\delta(q, x) = (q, 0, \blacktriangleright)$$

Question How do we construct a complete reversible aperiodic TM?



2. Cellular Automata

Cellular automata

Definition A **CA** is a triple (S, r, f) where S is a **finite set of states**, $r \in \mathbb{N}$ is the **radius** and $f : S^{2r+1} \to S$ is the **local rule** of the cellular automaton.

A **configuration** $c \in S^{\mathbb{Z}}$ is a coloring of $\mathbb{Z}$ by S.

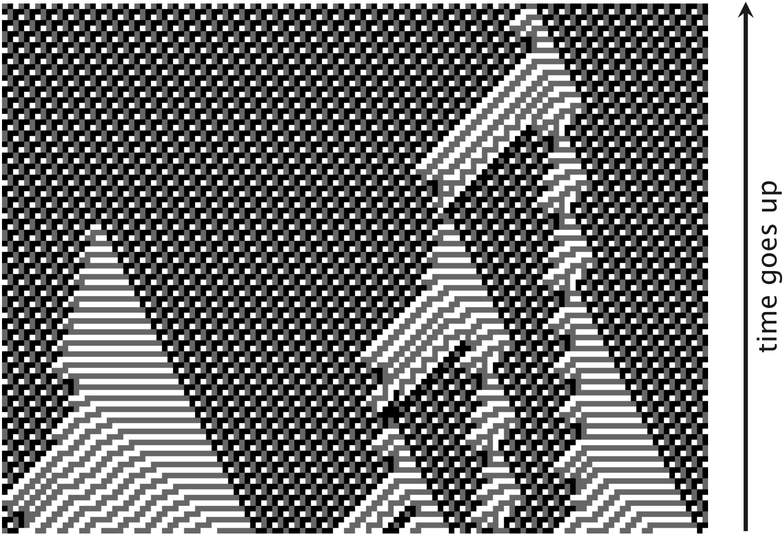
The **global map** $F : S^{\mathbb{Z}} \to S^{\mathbb{Z}}$ applies f uniformly and locally:

$$\forall c \in S^{\mathbb{Z}}, \forall z \in \mathbb{Z}, \qquad F(c)(z) = f(c(z-r), \ldots, c(z+r)).$$

A **space-time diagram** $\Delta \in S^{\mathbb{N}\times\mathbb{Z}}$ satisfies, for all $t \in \mathbb{Z}^+$,

$$\Delta(t+1) = F(\Delta(t)).$$

Space-time diagram



$S = \{0, 1, 2\}$, $r = 1$, $f(x, y, z) = \lfloor 6450288690466/3^{9x+3y+z} \rfloor \pmod 3$

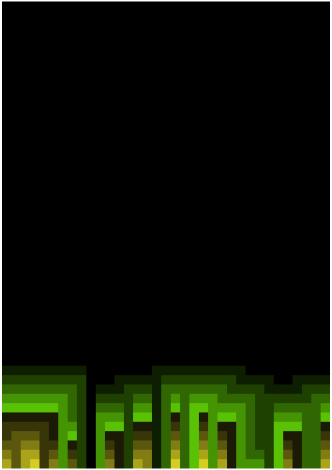
The nilpotency problem (Nil)

Definition A DDS is **nilpotent** if
$\exists z \in X, \forall x \in X, \exists n \in \mathbb{N}, F^n(x) = z.$

Given a recursive encoding of the DDS, can we **decide** nilpotency?

A DDS is **uniformly nilpotent** if
$\exists z \in X, \exists n \in \mathbb{N}, \forall x \in X, F^n(x) = z.$

Given a recursive encoding of the DDS, can we **bound recursively** n?



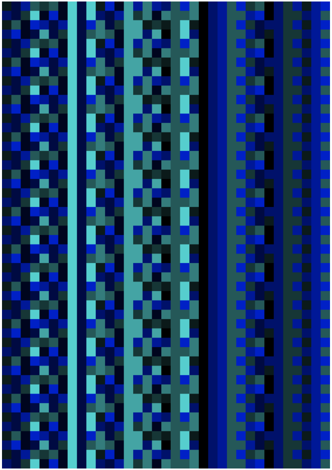
The periodicity problem (Per)

Definition A DDS is **periodic** if
$\forall x \in X, \exists n \in \mathbb{N}, F^n(x) = x$.

Given a recursive encoding of the DDS, can we **decide** periodicity?

A DDS is **uniformly periodic** if
$\exists n \in \mathbb{N}, \forall x \in X, F^n(x) = x$.

Given a recursive encoding of the DDS, can we **bound recursively** n?



Undecidability results

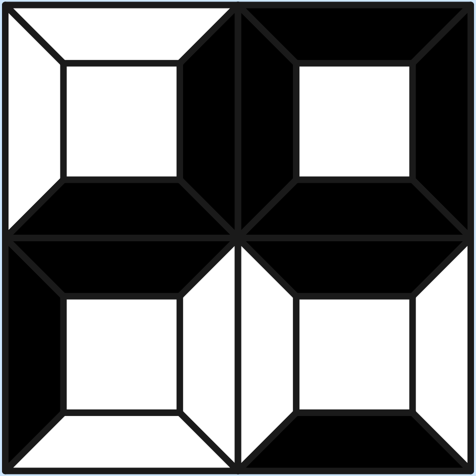
Theorem Both **Nil** and **Per** are **undecidable** for CA.

The proofs inject **computation** into **dynamics**.

Undecidability is not necessarily a negative result:
it is a **hint of complexity**.

Remark Due to **universe configurations** both nilpotency and periodicity are uniform.

The bounds grow **faster than any recursive function**: there exists simple nilpotent or periodic CA with huge bounds.

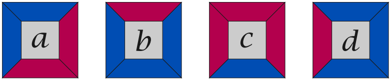


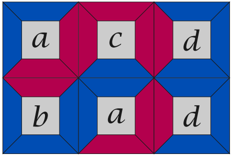
3. Nilpotency and tilings

The Domino Problem (DP)

*"Assume we are **given a finite set of square plates** of the same size with **edges colored**, each in a different manner. Suppose further there are infinitely many copies of each plate (plate type). We are **not permitted to rotate or reflect a plate**. The question is to find an effective procedure by which we can **decide**, for each given finite set of plates, **whether we can cover up the whole plane** (or, equivalently, an infinite quadrant thereof) **with copies of the plates subject to the restriction that adjoining edges must have the same color**."*

(Wang, 1961)





Undecidability of DP

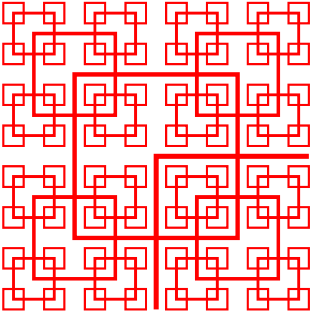
Theorem[Berger64] **DP** is **undecidable**.

Remark To prove it one needs **aperiodic** tile sets.

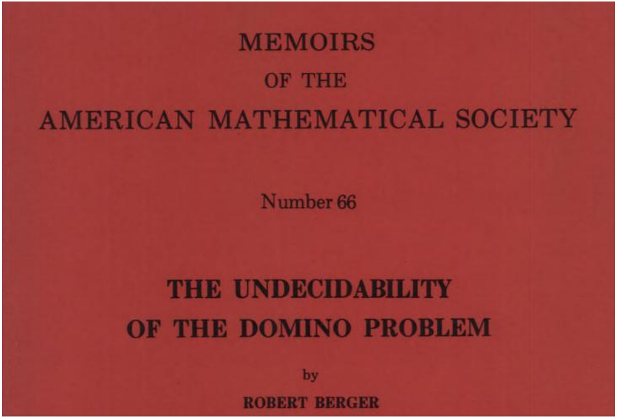
Idea of the proof

Enforce an (aperiodic) **self-similar structure** using local rules.

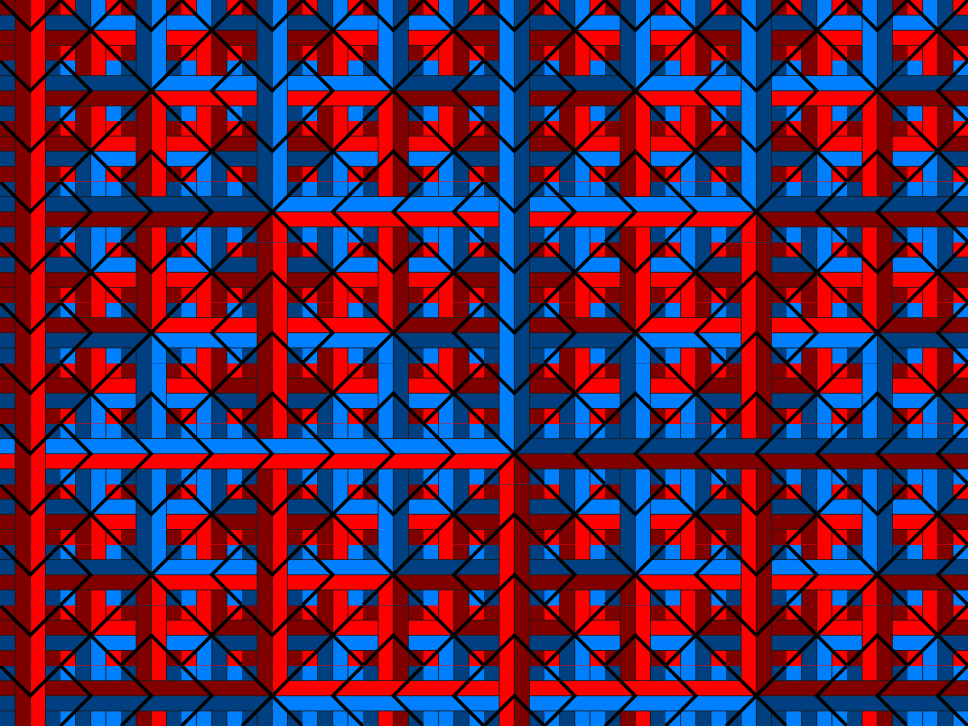
Insert a **Turing machine** computation **everywhere** using the structure.



Remark Plenty of different proofs!



"(. . .) In 1966 R. Berger discovered the first aperiodic tile set. It contains 20,426 Wang tiles, (. . .)
Berger himself managed to reduce the number of tiles to 104 and he described these in his thesis, though they were omitted from the published version (Berger [1966]). (. . .)" *[GrSh, p.584]*

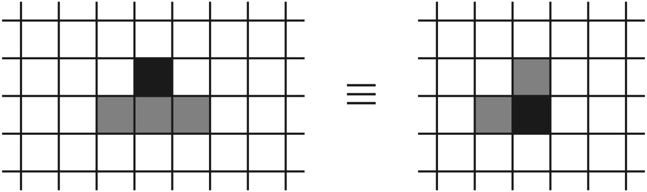


Reduction

A state $\perp \in S$ is **spreading** if $f(N) = \perp$ when $\perp \in N$.

A CA with a spreading state $\perp$ is not nilpotent iff it admits a biinfinite space-time diagram without $\perp$.

A tiling problem Find a coloring $\Delta \in (S \setminus \{\perp\})^{\mathbb{Z}^2}$ satisfying the tiling constraints given by f.



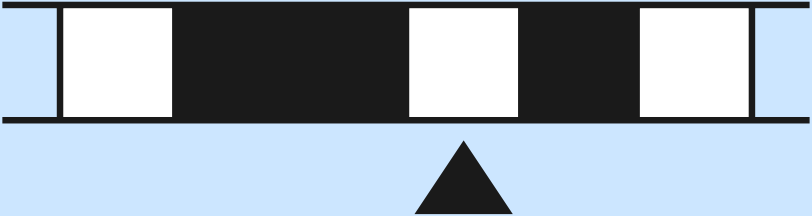
Theorem[Kari92] NW-DP $\leqslant_m$ Nil

Revisiting DP

Theorem[Kari92] **NW-DP** is **undecidable**.

Remark Reprove of undecidability of **DP** with the additionnal determinism constraint!

Corollary **Nil** is **undecidable**.



4. Periodicity and mortality

The Immortality Problem (IP)

“(T_2) To find an effective method, which for every Turing-machine M decides whether or not, for all tapes I (finite and infinite) and all states B, M will eventually halt if started in state B on tape I” *(Büchi, 1962)*

Definition A Turing machine is **mortal** if all configurations are ultimately halting.

Undecidability of IP

Theorem[Hooper66] **IP** is **undecidable**.

Remark To prove it one needs **aperiodic** TM.

Idea of the proof

Simulate 2-counters machines *à la* Minsky $(s, \underline{@}1^m x 2^n y)$

Replace **unbounded searches** by **recursive calls** to initial segments of the simulation.

Reduction: revisiting IP

Theorem[KO2008] **R-IP** $\leqslant_m$ **TM-Per** $\leqslant_m$ **Per**

Theorem[KO2008] **R-IP** is **undecidable**.

Remark Reprove of undecidability of **IP** with the additionnal reversibility constraint!

Immortality: a first attempt

"(T_2) To find an effective method, which for every Turing-machine M decides whether or not, for all tapes I (finite and infinite) and all states B, M will eventually halt if started in state B on tape I" *(Büchi, 1962)*

Immortality: a first attempt

"(T_2) *To find an effective method, which for every Turing-machine* M *decides whether or not, for all tapes* I *(finite and infinite) and all states* B, M *will eventually halt if started in state* B *on tape* I" *(Büchi, 1962)*

[Hooper66] IP is undecidable for TM.

Idea *TM with recursive calls! (we will discuss this)* ◊

Immortality: a first attempt

“(T_2) To find an effective method, which for every Turing-machine M decides whether or not, for all tapes I (finite and infinite) and all states B, M will eventually halt if started in state B on tape I” *(Büchi, 1962)*

[Hooper66] IP is undecidable for TM.
Idea *TM with recursive calls! (we will discuss this)* $\Diamond$

[Lecerf63] Every TM is **simulated** by a RTM.
Idea *Keep history on a stack encoded on the tape.* $\Diamond$

Immortality: a first attempt

"(T_2) *To find an effective method, which for every Turing-machine* M *decides whether or not, for all tapes* I *(finite and infinite) and all states* B, M *will eventually halt if started in state* B *on tape* I" *(Büchi, 1962)*

[Hooper66] IP is undecidable for TM.
Idea *TM with recursive calls! (we will discuss this)* ◊

[Lecerf63] Every TM is **simulated** by a RTM.
Idea *Keep history on a stack encoded on the tape.* ◊

Problem The simulation **does not** preserve immortality due to **unbounded searches**. We need to rewrite Hooper's proof for reversible machines.

Immortality: simulating RCM

Theorem 7 **IP** is undecidable for RTM.

Reduction reduce **HP** for 2-RCM $(s, \underline{@}1^m x 2^n y)$

Immortality: simulating RCM

Theorem 7 **IP** is undecidable for RTM.

Reduction reduce **HP** for 2-RCM $(s, \underline{@}1^m x2^n y)$

Problem unbounded searches produce immortality.
Idea *by compacity, extract infinite failure sequence*

Immortality: simulating RCM

Theorem 7 **IP** is undecidable for RTM.

Reduction reduce **HP** for 2-RCM $(s, \underline{@}1^m x 2^n y)$

Problem unbounded searches produce immortality.
Idea *by compacity, extract infinite failure sequence*

Hooper's trick use bounded searches with **recursive calls** to initial segments of the simulation of increasing sizes:

$\underset{s}{\underline{@}}$1111111111111x2222y *search x →*

Immortality: simulating RCM

Theorem 7 **IP** is undecidable for RTM.

Reduction reduce **HP** for 2-RCM $(s, \underline{@}1^m x2^n y)$

Problem unbounded searches produce immortality.
Idea *by compacity, extract infinite failure sequence*

Hooper's trick use bounded searches with **recursive calls** to initial segments of the simulation of increasing sizes:

$@\underset{\overline{s}'_1}{\underline{1}}111111111111x2222y$ *bounded search 1*

Immortality: simulating RCM

Theorem 7 **IP** is undecidable for RTM.

Reduction reduce **HP** for 2-RCM $(s, \underline{@}1^m x2^n y)$

Problem unbounded searches produce immortality.
Idea *by compacity, extract infinite failure sequence*

Hooper's trick use bounded searches with **recursive calls** to initial segments of the simulation of increasing sizes:

$@1\underset{\overline{s}_2'}{\underline{1}}11111111111x2222y$ *bounded search 2*

Immortality: simulating RCM

Theorem 7 **IP** is undecidable for RTM.

Reduction reduce **HP** for 2-RCM $(s, \underline{@}1^m x2^n y)$

Problem unbounded searches produce immortality.
Idea *by compacity, extract infinite failure sequence*

Hooper's trick use bounded searches with **recursive calls** to initial segments of the simulation of increasing sizes:

$@11\underset{\overline{s}'_3}{\underline{1}}11111111111x2222y$ *bounded search 3*

Immortality: simulating RCM

Theorem 7 **IP** is undecidable for RTM.

Reduction reduce **HP** for 2-RCM $(s, \underline{@}1^m x2^n y)$

Problem unbounded searches produce immortality.
Idea *by compacity, extract infinite failure sequence*

Hooper's trick use bounded searches with **recursive calls** to initial segments of the simulation of increasing sizes:

@$\underline{@}_s$**xy**111111111x2222y *recursive call*
$\overline{s}_0$

Immortality: simulating RCM

Theorem 7 **IP** is undecidable for RTM.

Reduction reduce **HP** for 2-RCM $(s, \underline{@}1^m x2^n y)$

Problem unbounded searches produce immortality.
Idea *by compacity, extract infinite failure sequence*

Hooper's trick use bounded searches with **recursive calls** to initial segments of the simulation of increasing sizes:

@**$@_s$11111x22222$\underset{s_c}{y}$**x2222y *ultimately in case of collision...*

Immortality: simulating RCM

Theorem 7 **IP** is undecidable for RTM.

Reduction reduce **HP** for 2-RCM $(s, \underline{@}1^m x 2^n y)$

Problem unbounded searches produce immortality.
Idea *by compacity, extract infinite failure sequence*

Hooper's trick use bounded searches with **recursive calls** to initial segments of the simulation of increasing sizes:

$@\underline{@}_{s}\mathbf{xy}1111111111x2222y$ *...revert to clean*
$\underline{\mathbf{s}}_{\mathbf{b}}$

Immortality: simulating RCM

Theorem 7 **IP** is undecidable for RTM.

Reduction reduce **HP** for 2-RCM $(s, \underline{@}1^m x2^n y)$

Problem unbounded searches produce immortality.
Idea *by compacity, extract infinite failure sequence*

Hooper's trick use bounded searches with **recursive calls** to initial segments of the simulation of increasing sizes:

$@111\underset{\overline{s}'_1}{\underline{1}}111111111x2222y$ *pop and continue bounded search 1*

Immortality: simulating RCM

Theorem 7 **IP** is undecidable for RTM.

Reduction reduce **HP** for 2-RCM $(s, \underline{@}1^m x2^n y)$

Problem unbounded searches produce immortality.
Idea *by compacity, extract infinite failure sequence*

Hooper's trick use bounded searches with **recursive calls** to initial segments of the simulation of increasing sizes:

$@1111\underset{\bar{s}_2'}{\underline{1}}11111111x2222y$ *bounded search 2*

Immortality: simulating RCM

Theorem 7 **IP** is undecidable for RTM.

Reduction reduce **HP** for 2-RCM $(s, \underline{@}1^m x2^n y)$

Problem unbounded searches produce immortality.
Idea *by compacity, extract infinite failure sequence*

Hooper's trick use bounded searches with **recursive calls** to initial segments of the simulation of increasing sizes:

$@11111\underset{\overline{s}_3'}{\underline{1}}1111111x2222y$ *bounded search 3*

Immortality: simulating RCM

Theorem 7 **IP** is undecidable for RTM.

Reduction reduce **HP** for 2-RCM $(s, \underline{@}1^m x 2^n y)$

Problem unbounded searches produce immortality.
Idea *by compacity, extract infinite failure sequence*

Hooper's trick use bounded searches with **recursive calls** to initial segments of the simulation of increasing sizes:

$@111\underset{s_0}{\underline{@}}_s\mathbf{xy}1111111x2222y$ *recursive call*

Immortality: simulating RCM

Theorem 7 **IP** is undecidable for RTM.

Reduction reduce **HP** for 2-RCM $(s, \underline{@}1^m x2^n y)$

Problem unbounded searches produce immortality.
Idea *by compacity, extract infinite failure sequence*

Hooper's trick use bounded searches with **recursive calls** to initial segments of the simulation of increasing sizes:

@111$\underline{@}_s$**xy**1111111x2222y *recursive call*
s_0

Immortality: simulating RCM

Theorem 7 **IP** is undecidable for RTM.

Reduction reduce **HP** for 2-RCM $(s, \underline{@}1^m x 2^n y)$

Problem unbounded searches produce immortality.
Idea *by compacity, extract infinite failure sequence*

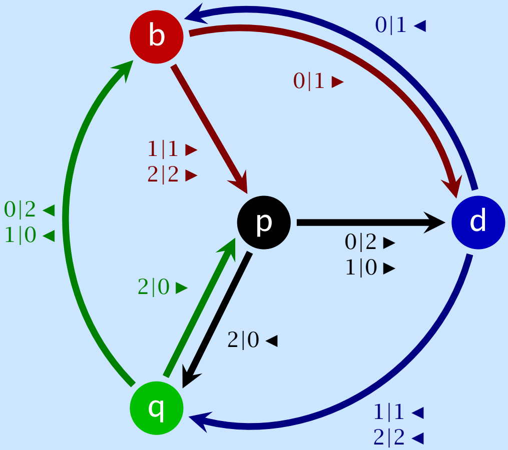
Hooper's trick use bounded searches with **recursive calls** to initial segments of the simulation of increasing sizes:

$@111\underset{s_0}{\underline{@}}_s xy1111111x2222y$ *recursive call*

The RTM is immortal iff the 2-RCM is mortal on $(s_0, (0, 0))$.

Program it!

```
def [s|search_1|t_0, t_1, t_2⟩ :
  s. @_α ⊢ @_α, l
  l. →, u
  u. x ⊢ x, t_0
   | 1x ⊢ 1x, t_1
   | 11x ⊢ 11x, t_2
   | 111 ⊢ 111, c
  call [c|check_1|p⟩ from 1
  p. 111 ⊢ 111, l

def [s|search_2|t_0, t_1, t_2⟩ :
  s. x ⊢ x, l
  l. →, u
  u. y ⊢ y, t_0
   | 2y ⊢ 2y, t_1
   | 22y ⊢ 22y, t_2
   | 222 ⊢ 222, c
  call [c|check_2|p⟩ from 2
  p. 222 ⊢ 222, l

def [s|test1|z, p⟩ :
  s. @_αx ⊢ @_αx, z
   | @_α1 ⊢ @_α1, p

def [s|endtest2|z, p⟩ :
  s. xy ⊢ xy, z
   | x2 ⊢ x2, p

def [s|test2|z, p⟩ :
  [s|search_1|t_0, t_1, t_2⟩
  [t_0|endtest2|z_0, p_0⟩
  [t_1|endtest2|z_1, p_1⟩
  [t_2|endtest2|z_2, p_2⟩
  ⟨z_0, z_1, z_2|search_1|z]
  ⟨p_0, p_1, p_2|search_1|p]

def [s|mark_1|t, co⟩ :
  s. y1 ⊢ 2y, t
   | yx ⊢ yx, co

def [s|endinc_1|t, co⟩ :
  [s|search_2|r_0, r_1, r_2⟩
  [r_0|mark_1|t_0, co_0⟩
  [r_1|mark_1|t_1, co_1⟩
  [r_2|mark_1|t_2, co_2⟩
  ⟨t_2, t_0, t_1|search_2|t]
  ⟨co_0, co_1, co_2|search_2|co]

def [s|inc2_1|t, co⟩ :
  [s|search_1|r_0, r_1, r_2⟩
  [r_0|endinc_1|t_0, co_0⟩
  [r_1|endinc_1|t_1, co_1⟩
  [r_2|endinc_1|t_2, co_2⟩
  ⟨t_0, t_1, t_2|search_1|t]
  ⟨co_0, co_1, co_2|search_1|co]

def [s|dec2_1|t⟩ :
  ⟨s, co|inc2_1|t]

def [s|mark_2|t, co⟩ :
  s. y2 ⊢ 2y, t
   | yx ⊢ yx, co

def [s|endinc_2|t, co⟩ :
  [s|search_2|r_0, r_1, r_2⟩
  [r_0|mark_2|t_0, co_0⟩
  [r_1|mark_2|t_1, co_1⟩
  [r_2|mark_2|t_2, co_2⟩
  ⟨t_2, t_0, t_1|search_2|t]
  ⟨co_0, co_1, co_2|search_2|co]

def [s|inc2_2|t, co⟩ :
  [s|search_1|r_0, r_1, r_2⟩
  [r_0|endinc_2|t_0, co_0⟩
  [r_1|endinc_2|t_1, co_1⟩
  [r_2|endinc_2|t_2, co_2⟩
  ⟨t_0, t_1, t_2|search_1|t]
  ⟨co_0, co_1, co_2|search_1|co]

def [s|dec2_2|t⟩ :
  ⟨s, co|inc2_2|t]

def [s|pushinc_1|t, co⟩ :
  s. x2 ⊢ 1x, c
   | xy1 ⊢ 1xy, pt
   | xyx ⊢ 1yx, pco
  [c|endinc_1|pt0, pco0⟩
  pt0. →, t0
  t0. 2 ⊢ 2, pt
  pt. ←, t
  pco0. x ⊢ 2, pco
  pco. ←, zco
  zco. 1 ⊢ x, co

def [s|inc1_1|t, co⟩ :
  [s|search_1|r_0, r_1, r_2⟩
  [r_0|pushinc_1|t_0, co_0⟩
  [r_1|pushinc_1|t_1, co_1⟩
  [r_2|pushinc_1|t_2, co_2⟩
  ⟨t_2, t_0, t_1|search_1|t]
  ⟨co_0, co_1, co_2|search_1|co]

def [s|dec1_1|t⟩ :
  ⟨s, co|inc1_1|t]

def [s|pushinc_2|t, co⟩ :
  s. x2 ⊢ 1x, c
   | xy2 ⊢ 1xy, pt
   | xyy ⊢ 1yy, pco
  [c|endinc_2|pt0, pco0⟩
  pt0. →, t0
  t0. 2 ⊢ 2, pt
  pt. ←, t
  pco0. x ⊢ 2, pco
  pco. ←, zco
  zco. 1 ⊢ x, co

def [s|inc1_2|t, co⟩ :
  [s|search_1|r_0, r_1, r_2⟩
  [r_0|pushinc_2|t_0, co_0⟩
  [r_1|pushinc_2|t_1, co_1⟩
  [r_2|pushinc_2|t_2, co_2⟩
  ⟨t_2, t_0, t_1|search_1|t]
  ⟨co_0, co_1, co_2|search_1|co]

def [s|dec1_2|t⟩ :
  ⟨s, co|inc1_2|t]

def [s|init_1|r⟩ :
  s. →, u
  u. 11 ⊢ xy, e
  e. ←, r

def [s|RCM_1|co_1, co_2⟩ :
  [s|init_1|s_0⟩
  [s_0|test1|s_1z, n⟩
  [s_1|inc1_1|s_2, co_1⟩
  [s_2|inc2_1|s_3, co_2⟩
  [s_3|test1|n', s_1p⟩
  ⟨s_1z, s_1p|test1|s_1]

def [s|init_2|r⟩ :
  s. →, u
  u. 22 ⊢ xy, e
  e. ←, r

def [s|RCM_2|co_1, co_2⟩ :
  [s|init_2|s_0⟩
  [s_0|test1|s_1z, n⟩
  [s_1|inc1_2|s_2, co_1⟩
  [s_2|inc2_2|s_3, co_2⟩
  [s_3|test1|n', s_1p⟩
  ⟨s_1z, s_1p|test1|s_1]

fun [s|check_1|t⟩ :
  [s|RCM_1|co_1, co_2, ...⟩
  ⟨co_1, co_2, ...|RCM_1|t]

fun [s|check_2|t⟩ :
  [s|RCM_2|co_1, co_2, ...⟩
  ⟨co_1, co_2, ...|RCM_2|t]
```



5. a SMART machine

The SMART machine $\mathfrak{C}$

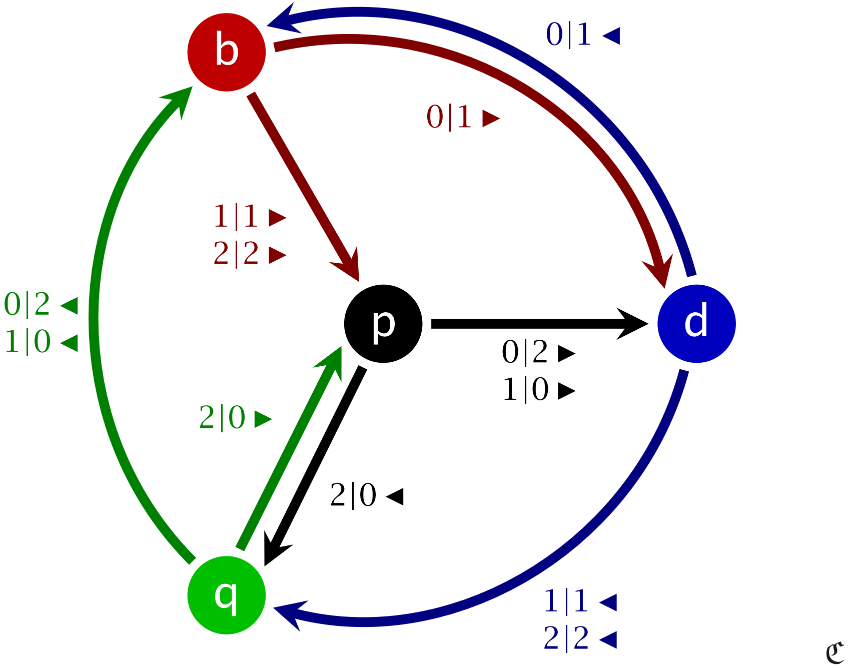
Conj[Kůrka97] Every **complete** TM has a **periodic** point.

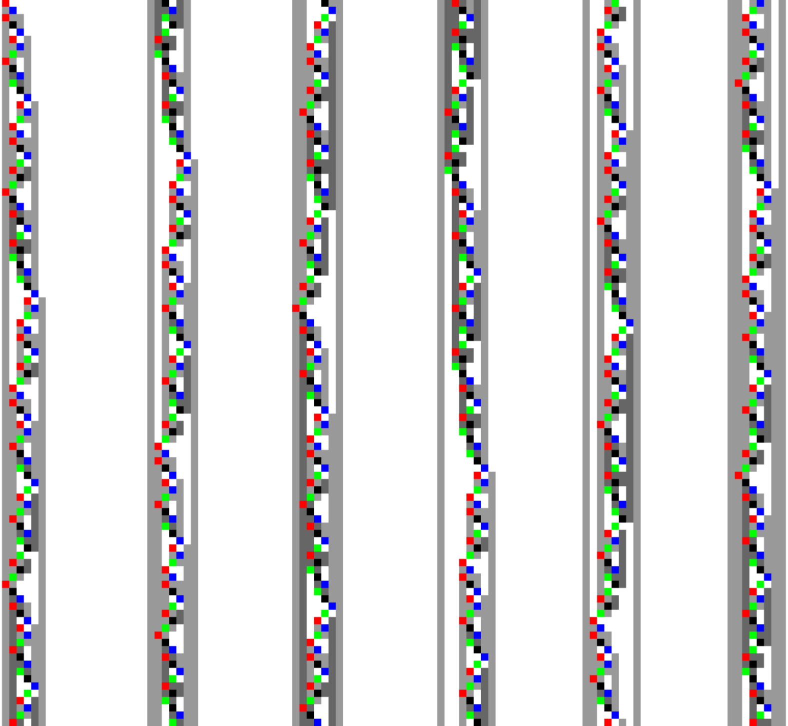
Thm[BCN02] No, here is an **aperiodic** complete TM.

Rk It relies on the **bounded search** technique [Hooper66].

In 2008, I asked **J. Cassaigne** if he had a reversible version of the BCN construction...

...he answered with a small machine $\mathfrak{C}$ which is a reversible and (drastic) simplification of the BCN machine.


b
0|1 ◀
0|1 ▶
1|1 ▶
2|2 ▶
0|2 ◀
1|0 ◀
p
d
0|2 ▶
1|0 ▶
2|0 ▶
2|0 ◀
q
1|1 ◀
2|2 ◀



The SMART machine $\mathfrak{C}$

A 4-state 3-symbols TM with nice properties:

complete no halting configuration

reversible reversed by a TM...

time-symetric ...essentially itself (up to details)

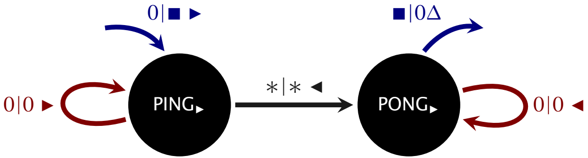
aperiodic no time periodic orbit

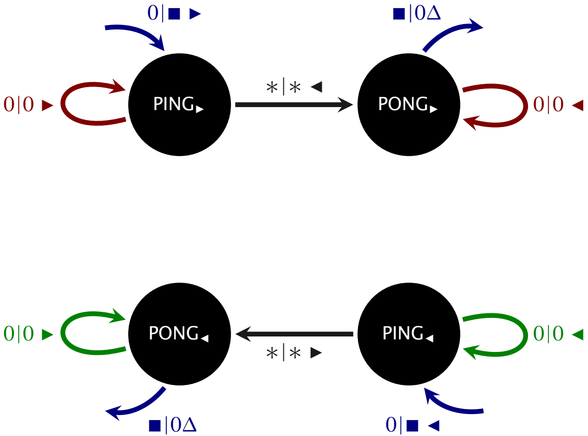
substitutive substitution-generated trace-shift language

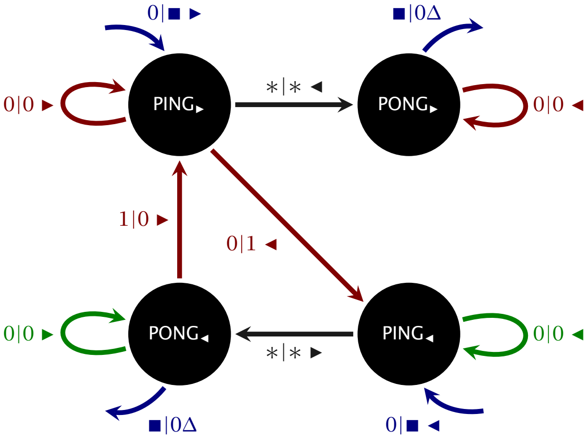
TMH-transitive dense orbits with moving head

TMT-minimal every orbit is dense with moving tape

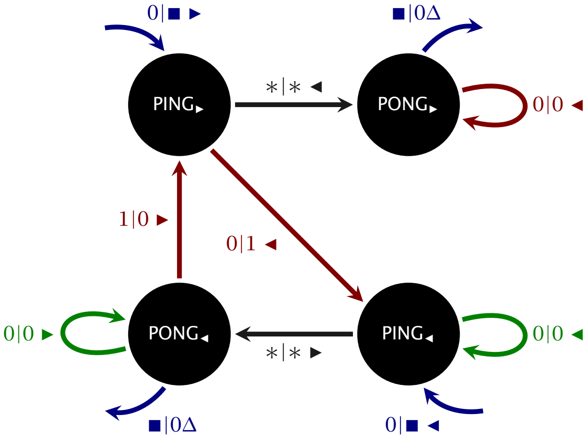
How does it work?


0|■ ▶
■|0Δ
0|0 ▶
PING▸
∗|∗ ◀
PONG▸
0|0 ◀

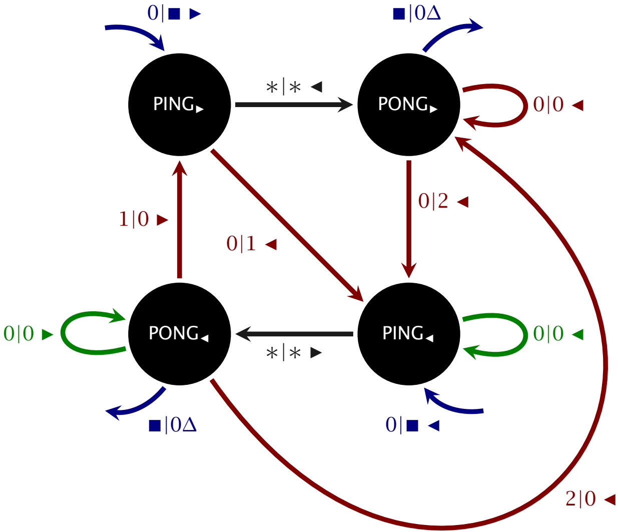

0|■ ▶
■|0Δ
0|0 ▶
PING▶
∗|∗ ◀
PONG▶
0|0 ◀
0|0 ▶
PONG◀
PING◀
0|0 ◀
∗|∗ ▶
■|0Δ
0|■ ◀

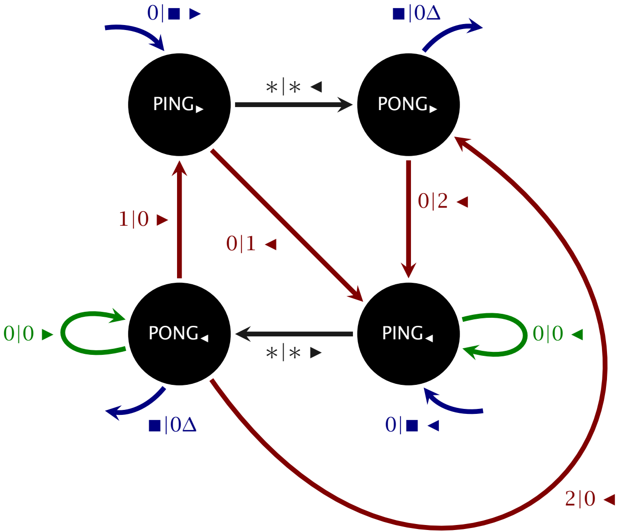


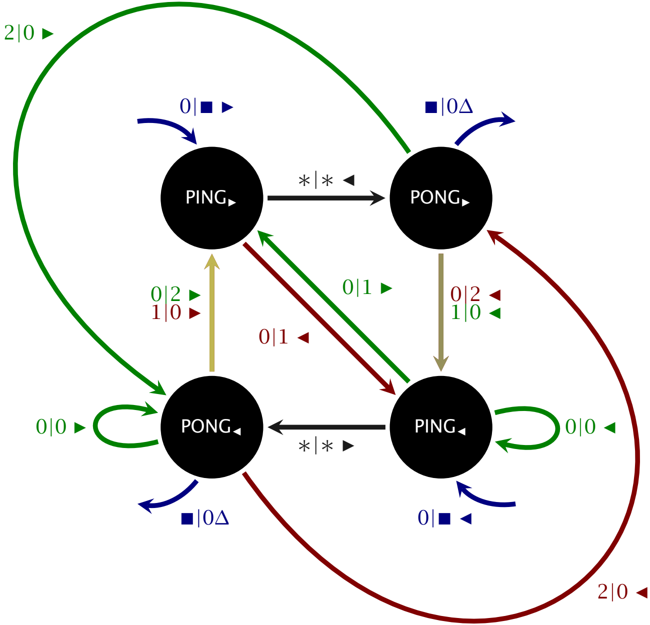
0|■ ▶
■|0Δ
0|0 ▶
PING▸
∗|∗ ◀
PONG▸
0|0 ◀
1|0 ▶
0|1 ◀
0|0 ▶
PONG◂
PING◂
0|0 ◀
∗|∗ ▶
■|0Δ
0|■ ◀

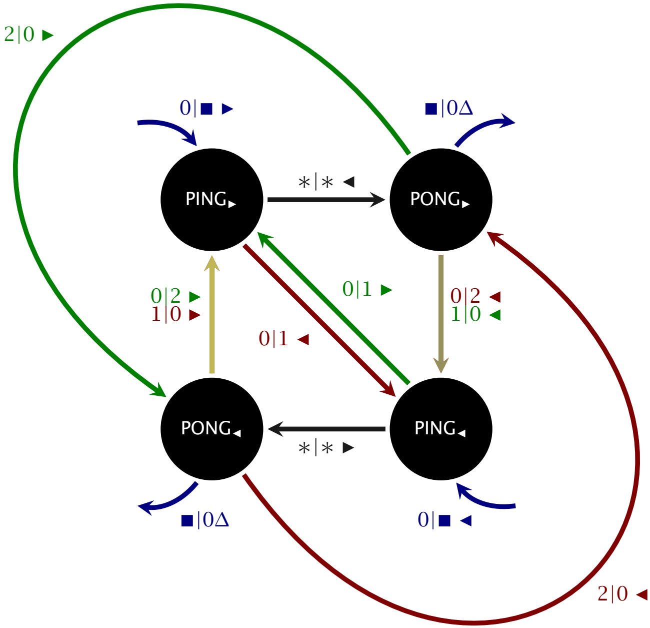


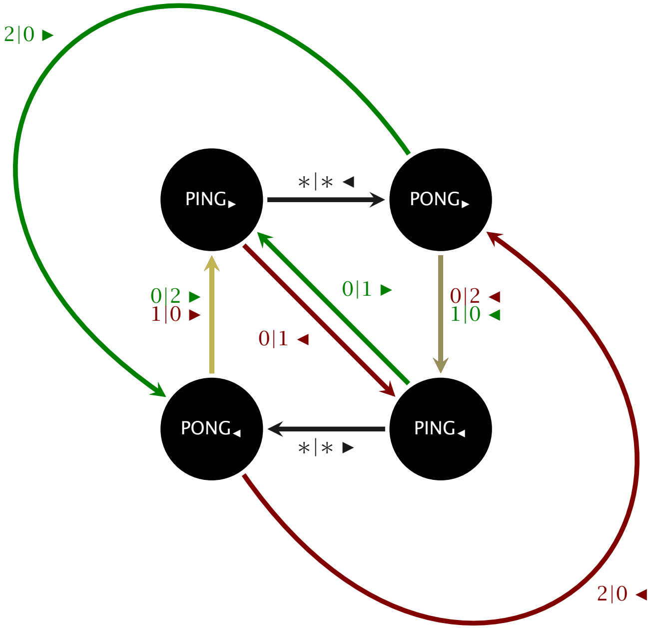
0|■ ▶
■|0Δ
| ◀
PING▸
PONG▸
0|0 ◀
1|0 ▶
0|1 ◀
0|0 ▶
PONG◂
PING◂
0|0 ◀
| ▶
■|0Δ
0|■ ◀


0|■ ▶
■|0Δ
PING▶
| ◀
PONG▶
0|0 ◀
0|2 ◀
1|0 ▶
0|1 ◀
0|0 ▶
PONG◀
PING◀
0|0 ◀
| ▶
■|0Δ
0|■ ◀
2|0 ◀

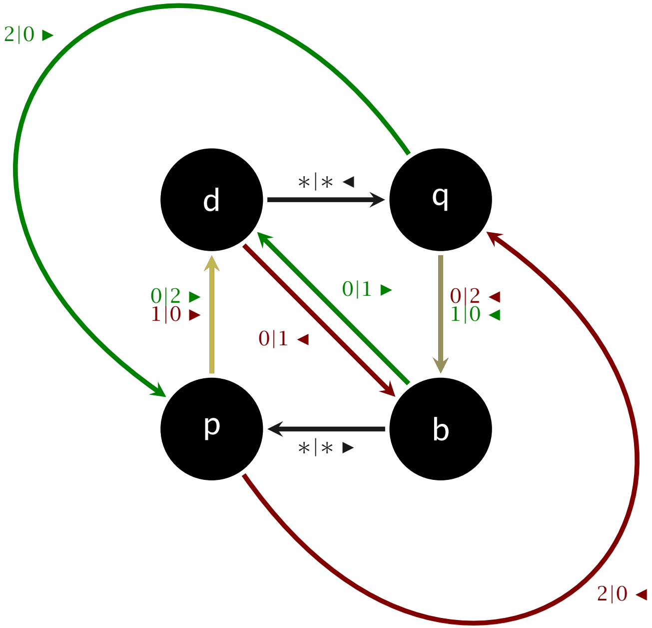

0|■ ▶
■|0Δ
PING▸
∗|∗ ◀
PONG▸
0|2 ◀
1|0 ▶
0|1 ◀
0|0 ▶
PONG◂
PING◂
0|0 ◀
∗|∗ ▶
■|0Δ
0|■ ◀
2|0 ◀


2|0 ▶
0|■ ▶
■|0Δ
∗|∗ ◀
PING▶
PONG▶
0|1 ▶
0|2 ▶
1|0 ▶
0|2 ◀
1|0 ◀
0|1 ◀
0|0 ▶
PONG◀
PING◀
0|0 ◀
∗|∗ ▶
■|0Δ
0|■ ◀
2|0 ◀


2|0 ▶
0|■ ▶
■|0Δ
PING▸
∗|∗ ◀
PONG▸
0|1 ▶
0|2 ▶
1|0 ▶
0|1 ◀
0|2 ◀
1|0 ◀
PONG◂
∗|∗ ▶
PING◂
■|0Δ
0|■ ◀
2|0 ◀



2|0 ▶
PING▶
∗|∗ ◀
PONG▶
0|2 ▶
1|0 ▶
0|1 ▶
0|1 ◀
0|2 ◀
1|0 ◀
PONG◀
PING◀
∗|∗ ▶
2|0 ◀


2|0 ▶
d
∗|∗ ◀
q
0|2 ▶
1|0 ▶
0|1 ▶
0|1 ◀
0|2 ◀
1|0 ◀
p
b
∗|∗ ▶
2|0 ◀

Recursive behavior

PING$_\blacktriangleright(n)$:
    for i=1 to n:
        d. 0|1, b ◀
        PING$_\blacktriangleleft(i-1)$
    d. $x|x$, q ◀
    for i=n downto 1:
        q. 0|2, b ◀
        PING$_\blacktriangleleft(i-1)$
    q. $y|0$, $\alpha(y)$ $\tau(y)$

PING$_\blacktriangleleft(n)$:
    for i=1 to n:
        b. 0|1, d ▶
        PING$_\blacktriangleright(i-1)$
    b. $x|x$, p ▶
    for i=n downto 1:
        p. 0|2, d ▶
        PING$_\blacktriangleright(i-1)$
    p. $y|0$, $\alpha'(y)$ $\tau'(y)$

$$\begin{cases} f(0) & = 2 \\ f(n+1) & = 3f(n) \end{cases}$$

Substitutive trace subshift

$$\varphi\begin{pmatrix}0\\ \mathbf{b}\end{pmatrix} = \begin{matrix}0 & 0 & 1 & 1\\ \mathbf{b} & \mathbf{d} & \mathbf{b} & \mathbf{p}\end{matrix} \qquad \varphi\begin{pmatrix}0\\ \mathbf{d}\end{pmatrix} = \begin{matrix}0 & 0 & 1 & 1\\ \mathbf{d} & \mathbf{b} & \mathbf{d} & \mathbf{q}\end{matrix}$$

$$\varphi\begin{pmatrix}\mathrm{x}\\ \mathbf{b}\end{pmatrix} = \begin{matrix}\mathrm{x}\\ \mathbf{b}\end{matrix} \qquad \varphi\begin{pmatrix}\mathrm{x}\\ \mathbf{d}\end{pmatrix} = \begin{matrix}\mathrm{x}\\ \mathbf{d}\end{matrix}$$

$$\varphi\begin{pmatrix}0\\ \mathbf{p}\end{pmatrix} = \begin{matrix}0 & 0 & 2 & 1\\ \mathbf{p} & \mathbf{d} & \mathbf{b} & \mathbf{p}\end{matrix} \qquad \varphi\begin{pmatrix}0\\ \mathbf{q}\end{pmatrix} = \begin{matrix}0 & 0 & 2 & 1\\ \mathbf{q} & \mathbf{b} & \mathbf{d} & \mathbf{q}\end{matrix}$$

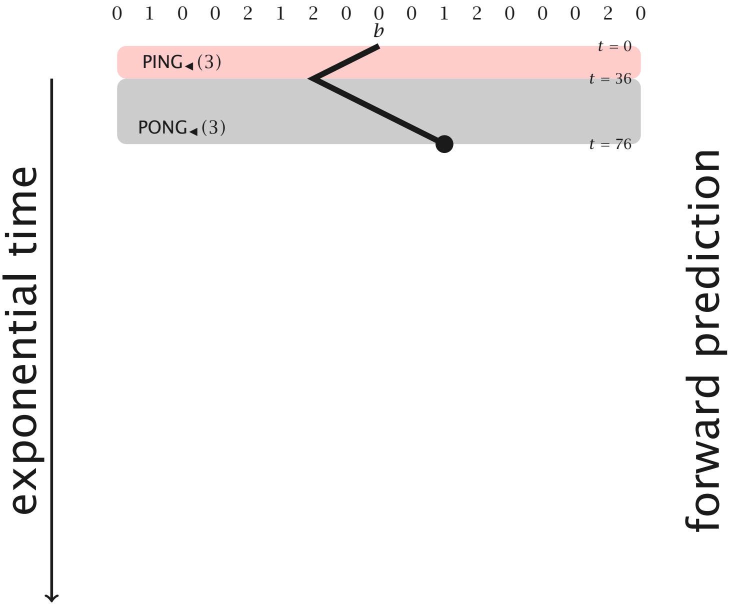
$$\varphi\begin{pmatrix}\mathrm{x}\\ \mathbf{p}\end{pmatrix} = \begin{matrix}0 & \mathrm{x} & 2 & \mathrm{x}\\ \mathbf{p} & \mathbf{d} & \mathbf{q} & \mathbf{p}\end{matrix} \qquad \varphi\begin{pmatrix}\mathrm{x}\\ \mathbf{q}\end{pmatrix} = \begin{matrix}0 & \mathrm{x} & 2 & \mathrm{x}\\ \mathbf{q} & \mathbf{b} & \mathbf{p} & \mathbf{q}\end{matrix}$$

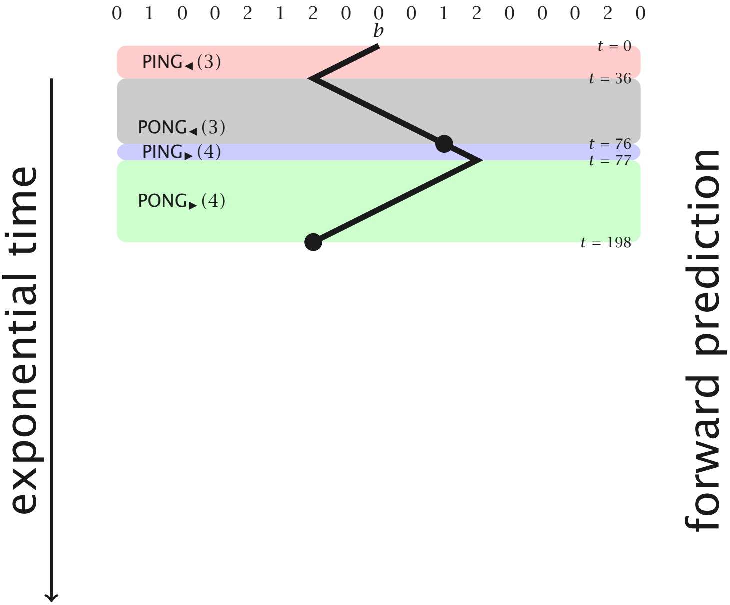
0 1 0 0 2 1 2 0 0 0 1 2 0 0 0 2 0

b

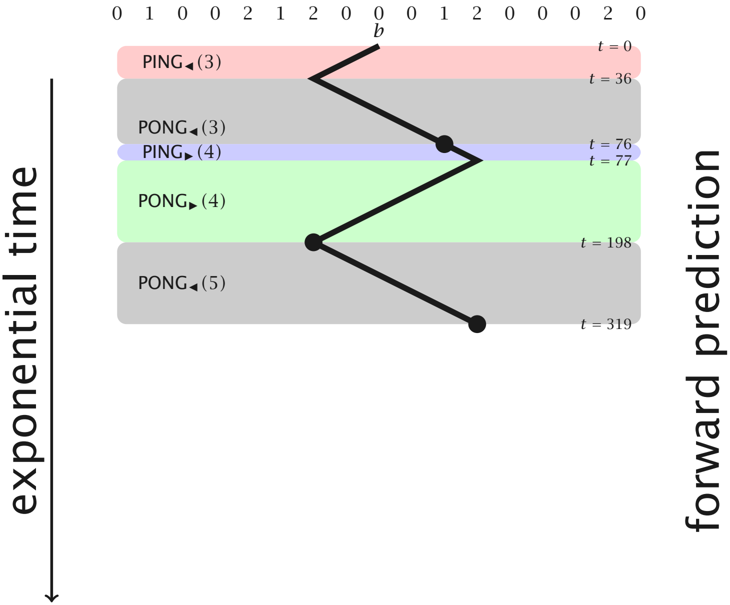
exponential time

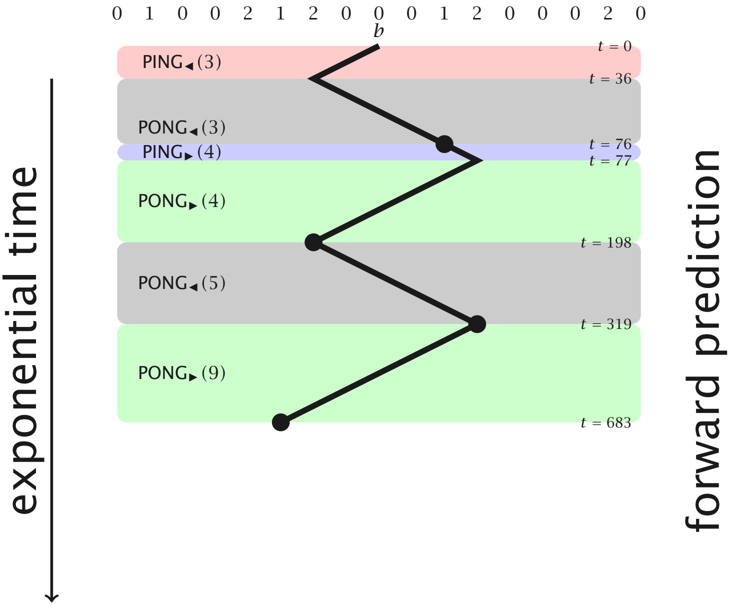
forward prediction

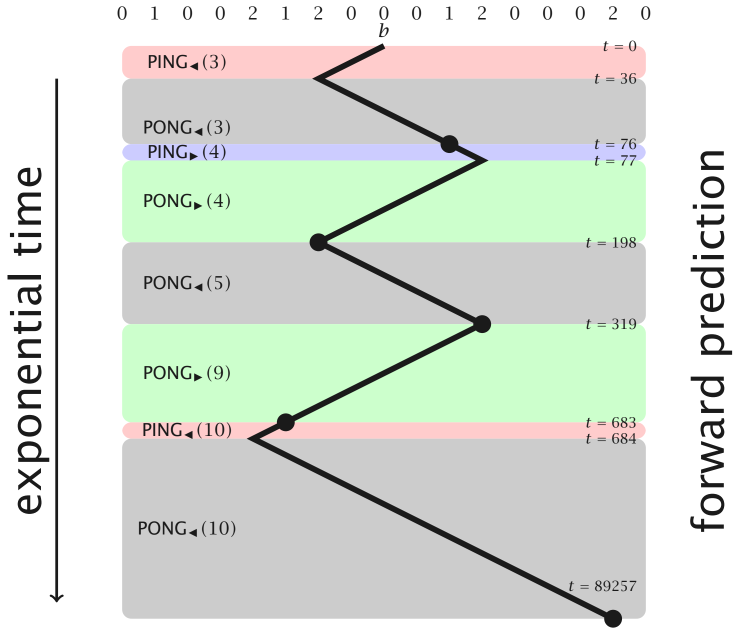

0 1 0 0 2 1 2 0 0 0 1 2 0 0 0 2 0
b
t = 0
PING◀(3)
t = 36
PONG◀(3)
t = 76
exponential time
forward prediction

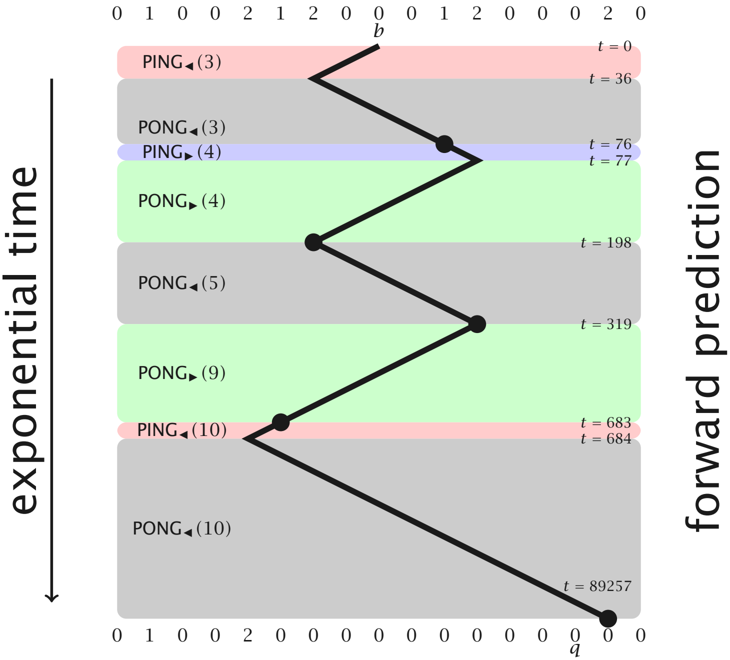


0 1 0 0 2 1 2 0 0 0 1 2 0 0 0 2 0
b
t = 0
PING◀(3)
t = 36
PONG◀(3)
t = 76
PING▶(4)
t = 77
PONG▶(4)
t = 198
exponential time
forward prediction

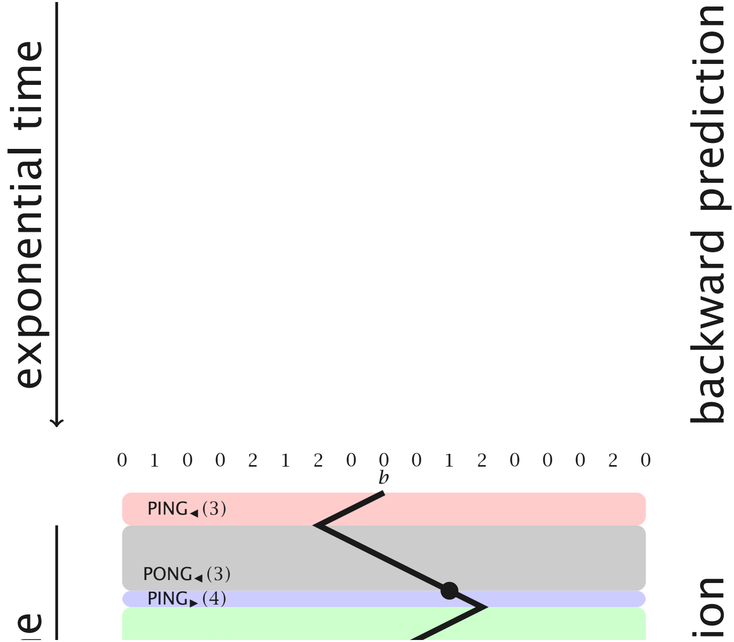

0 1 0 0 2 1 2 0 0 0 1 2 0 0 0 2 0
b
t = 0
PING◀(3)
t = 36
PONG◀(3)
t = 76
PING▶(4)
t = 77
PONG▶(4)
t = 198
PONG◀(5)
t = 319
exponential time
forward prediction


0 1 0 0 2 1 2 0 0 0 1 2 0 0 0 2 0
b
t = 0
PING◀(3)
t = 36
PONG◀(3)
t = 76
PING▶(4)
t = 77
PONG▶(4)
t = 198
PONG◀(5)
t = 319
PONG▶(9)
t = 683
exponential time
forward prediction

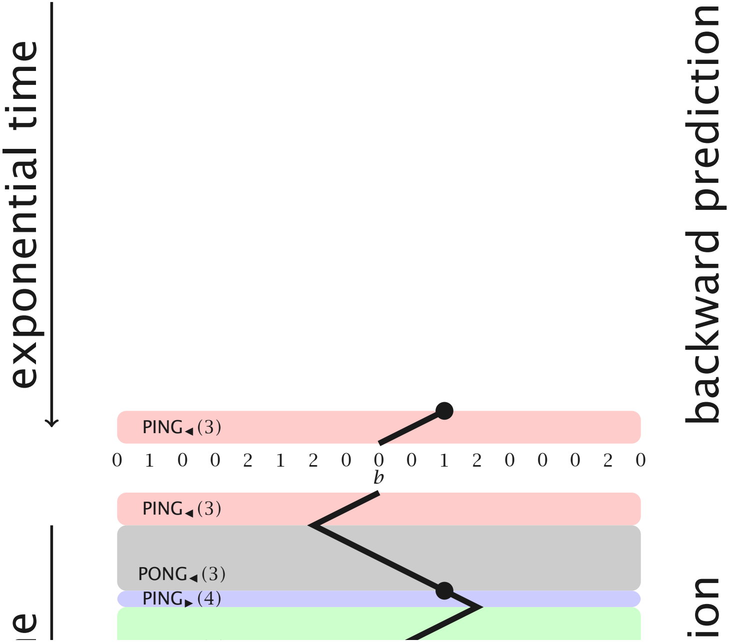

0 1 0 0 2 1 2 0 0 0 1 2 0 0 0 2 0
b
t = 0
PING◀(3)
t = 36
PONG◀(3)
t = 76
PING▶(4)
t = 77
PONG▶(4)
t = 198
PONG◀(5)
t = 319
PONG▶(9)
t = 683
PING◀(10)
t = 684
PONG◀(10)
t = 89257
exponential time
forward prediction

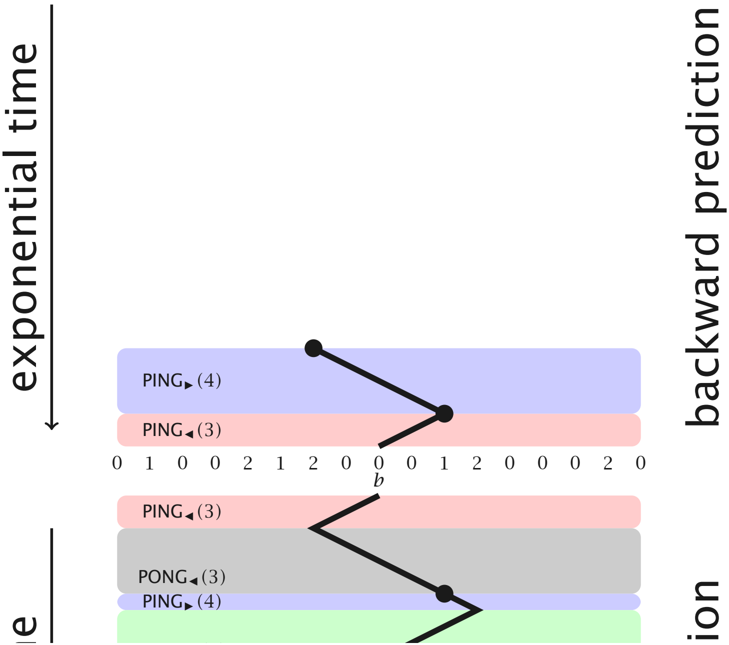


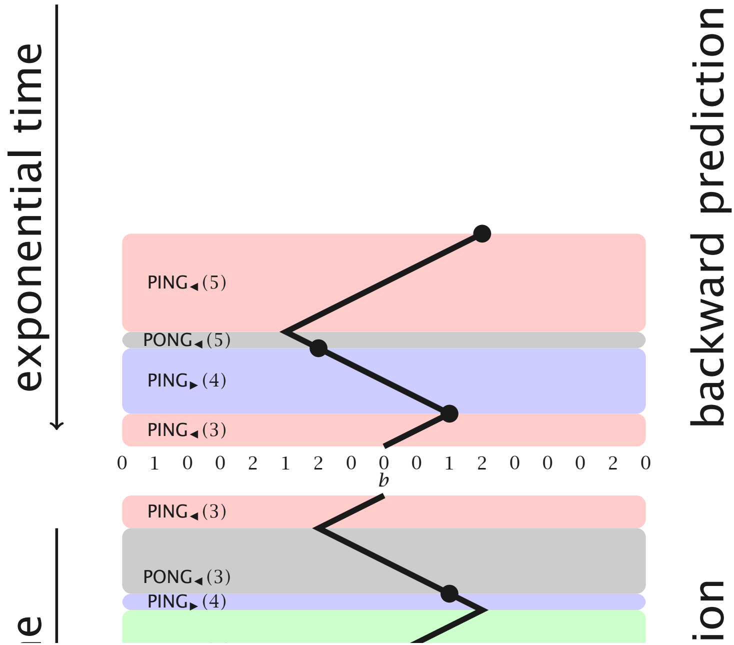
0 1 0 0 2 1 2 0 0 0 1 2 0 0 0 2 0
b
t = 0
PING◂(3)
t = 36
PONG◂(3)
t = 76
PING▸(4)
t = 77
PONG▸(4)
t = 198
PONG◂(5)
t = 319
PONG▸(9)
t = 683
PING◂(10)
t = 684
PONG◂(10)
t = 89257
0 1 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0
q
exponential time
forward prediction

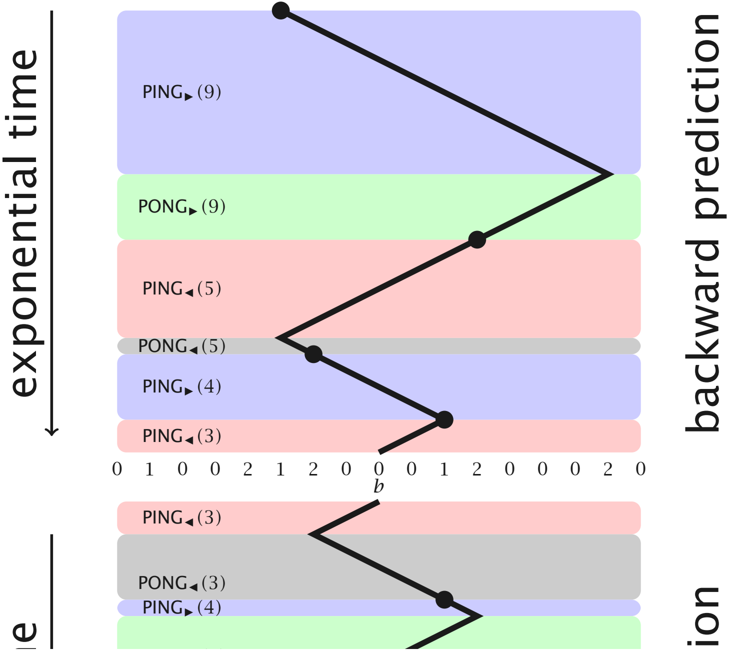


exponential time
backward prediction
0 1 0 0 2 1 2 0 0 0 1 2 0 0 0 2 0
b
PING◀(3)
PONG◀(3)
PING▶(4)

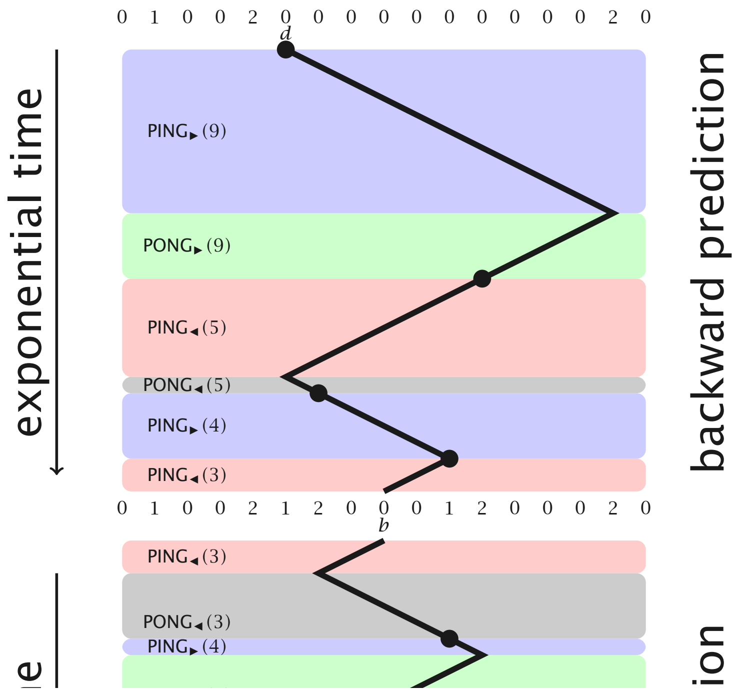

exponential time
backward prediction
PING◂(3)
0 1 0 0 2 1 2 0 0 0 1 2 0 0 0 2 0
b
PING◂(3)
PONG◂(3)
PING▸(4)
ion


exponential time
backward prediction
PING▸(4)
PING◂(3)
0 1 0 0 2 1 2 0 0 0 1 2 0 0 0 2 0
b
PING◂(3)
PONG◂(3)
PING▸(4)


exponential time
backward prediction
PING◂(5)
PONG◂(5)
PING▸(4)
PING◂(3)
0 1 0 0 2 1 2 0 0 0 1 2 0 0 0 2 0
b
PING◂(3)
PONG◂(3)
PING▸(4)
ion



exponential time
backward prediction
PING▸ (9)
PONG▸ (9)
PING◂ (5)
PONG◂ (5)
PING▸ (4)
PING◂ (3)
0 1 0 0 2 1 2 0 0 0 1 2 0 0 0 2 0
b
PING◂ (3)
PONG◂ (3)
PING▸ (4)



0 1 0 0 2 0 0 0 0 0 0 0 0 0 0 2 0
d
exponential time
backward prediction
PING▸ (9)
PONG▸ (9)
PING◂ (5)
PONG◂ (5)
PING▸ (4)
PING◂ (3)
0 1 0 0 2 1 2 0 0 0 1 2 0 0 0 2 0
b
PING◂ (3)
PONG◂ (3)
PING▸ (4)
ion

SMART is (TMH-)transitive

Proposition $\left({}^{\omega}2 \cdot \underset{p}{2}\ 2^{\omega} \right)$ is a **transitive point**.

Proof

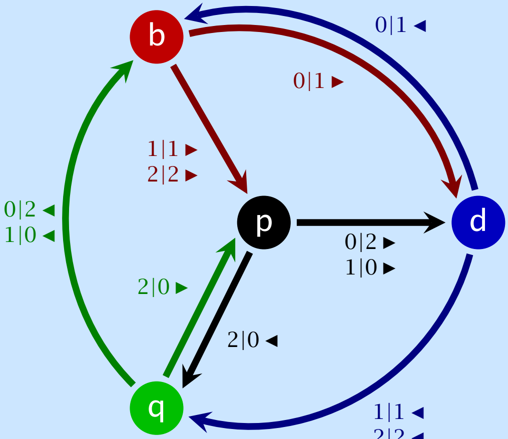
(Forward) For all $k \geqslant 0$:

$$\left({}^{\omega}2 \cdot \underset{p}{2}\ 2^{\omega} \right) \vdash^{*} \left({}^{\omega}2\ \underset{q}{2}\ 0^{k} \cdot 0\ 0^{k}\ 2^{\omega} \right) \quad .$$

(Backward) For every partial configuration $\left(\underset{\leftarrow}{u}\ \underset{\alpha}{\cdot}\ \underset{\rightarrow}{v} \right)$, there exist $w, w' \in \{0,1,2\}^{*}$ and $k > 0$ big enough such that

$$\left({}^{\omega}2\ \underset{q}{2}\ 0^{k} \cdot 0\ 0^{k}\ 2^{\omega} \right) \vdash^{*} \left({}^{\omega}2\ w\ \underset{\leftarrow}{u}\ \underset{\alpha}{\cdot}\ \underset{\rightarrow}{v}\ w'\ 2^{\omega} \right) \quad .$$

■



6. The embedding technique

Searching for a reduction

If we want to prove the following:

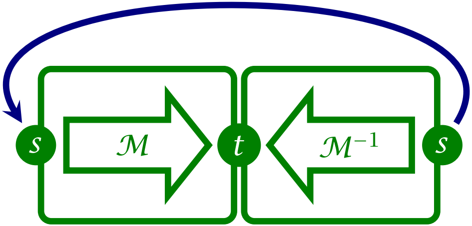
Theorem To find if a given **complete reversible Turing machine** admits a **periodic orbit** is Σ_1-complete.

In the partial case we use the following tool:

Prop[KO08] To find if a given **(aperiodic) RTM** can reach a given state t from a given state s is Σ_1-complete.

The partial case

Principle of the reduction Associate to an (aperiodic) RTM $\mathcal{M}$ with given s and t a new machine with a periodic orbit if and only if t is reachable from s.



We need to find a way to **complete** the constructed machine.
We will **embed** it into a **complete aperiodic** RTM.

Reversing time

Combine Turing machines to construct bigger ones.

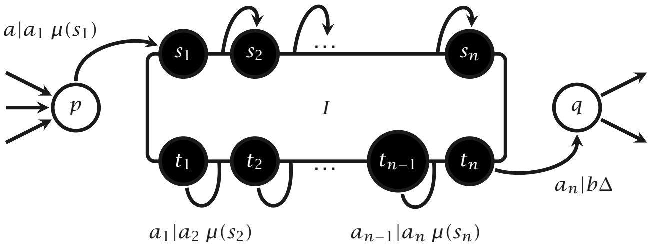
Reversing the time Given a reversible TM $M = (Q, \Sigma, \delta)$, construct $M_+ = (Q \times \{+\}, \Sigma, \delta^+)$ and $M_- = (Q \times \{-\}, \Sigma, \delta^-)$ where $(s, +)$ encodes M in state s running **forward** and $(s, -)$ running **backward**.

A typical use connects halting pairs from one machine to the corresponding starting pair of the other.

Embedding technique

A TM I with starting pairs $(s_1, a_1), \ldots, (s_n, a_n)$ and halting pairs $(t_1, a_1), \ldots, (t_n, a_n)$ is **innocuous** if starting from $(s_i, c, p + \mu(s_i))$ where $c(p) = a_i$ the machine might only halt in (t_i, c, p).

The **embedding** H^I of an **invited** innocuous TM I inside a **host** TM H is the TM containing a copy of both I and H where one transition $\delta(p, a) = (q, b, \Delta)$ from H is replaced by



Undecidability of transitivity

BRA Reachability Problem[Σ_1^0-comp. too] Given a binary reversible aperiodic TM, a starting pair (s, a) and a halting pair (t, b), decide if (t, b) is reachable from (s, a).

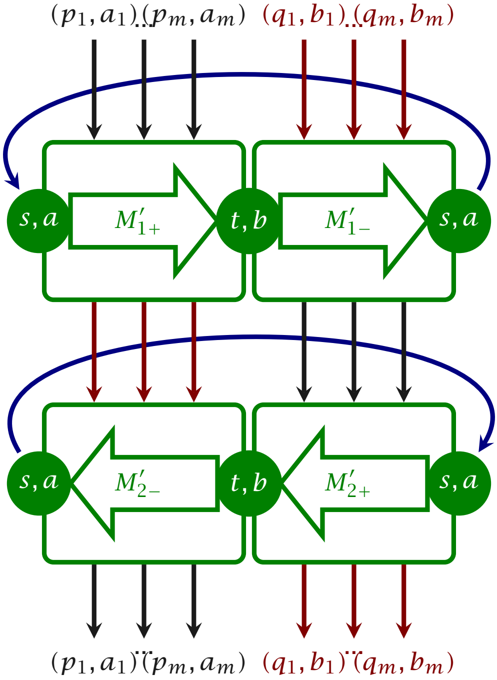
Theorem $\overline{\text{BRA Reachability Problem}} \leqslant_m$ Transitivity Problem

Proof

Let M, (s, a), (t, b) be an instance of the BRA Reachability Problem and M' be a copy of M with a third symbol $\$$.

Apply *Reversing time* to 2 copies of M' to construct an innocuous TM I as follows.

SMART^I is transitive iff (t, b) is not reachable from (s, a). ■



$(p_1, a_1) \dots (p_m, a_m)$
$(q_1, b_1) \dots (q_m, b_m)$
s, a
M'_{1+}
t, b
M'_{1-}
s, a
s, a
M'_{2-}
t, b
M'_{2+}
s, a
$(p_1, a_1) \dots (p_m, a_m)$
$(q_1, b_1) \dots (q_m, b_m)$

Conclusion

The embedding technique can be used to prove several undecidability results on TM.

Theorem **Transitivity** is Π_1^0-hard in TMH, TMT and ST.

Theorem **Minimality** is Σ_1^0-hard in TMT and ST.

Theorem To find if a given **complete reversible Turing machine** admits a **periodic orbit** is Σ_1-complete.

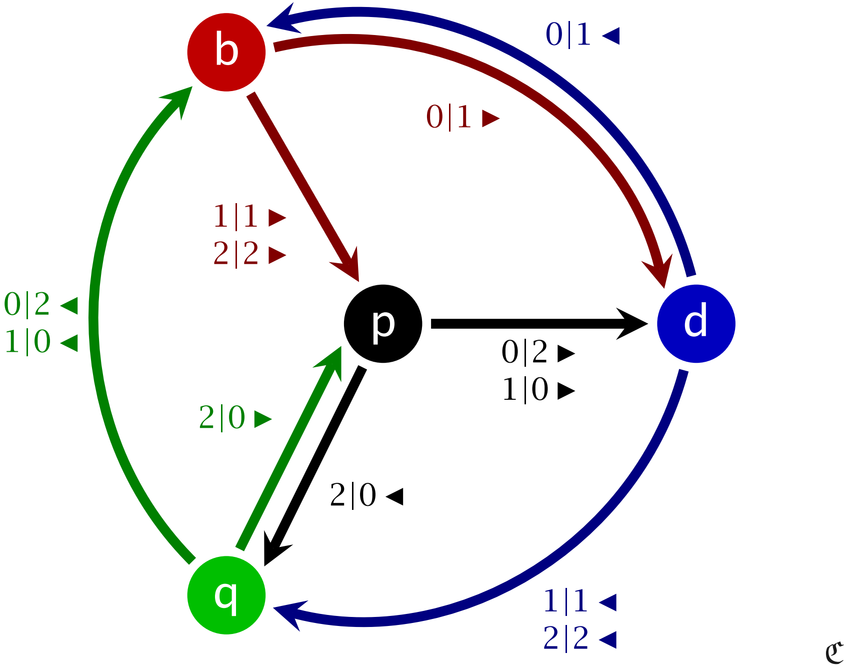

b
p
d
q
0|1 ◀
0|1 ▶
1|1 ▶
2|2 ▶
0|2 ◀
1|0 ◀
0|2 ▶
1|0 ▶
2|0 ▶
2|0 ◀
1|1 ◀
2|2 ◀
𝔈

Table of contents

1. Dynamics of Turing machines

2. Cellular Automata

3. Nilpotency and tilings

4. Periodicity and mortality

5. a SMART machine

6. The embedding technique