IMA Summer Program on
Codes, Systems and Graphical Models
August 1999

Asynchronous sliding block maps

M.-P. Béal
O. Carton

Institut Gaspard Monge
Université de Marne-la-Vallée

http://www-igm.univ-mlv.fr/~beal/
http://www-igm.univ-mlv.fr/~carton/

A alphabet

bi-infinite word: $x = \ldots a_{-2}a_{-1}a_0a_1a_2\ldots$

${}^\omega A^\omega$ set of bi-infinite words over A.

Sliding block map:

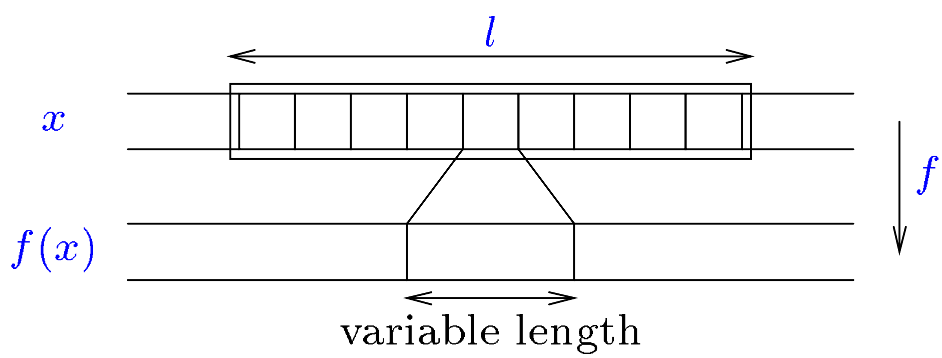


Figure 1: Sliding block map f

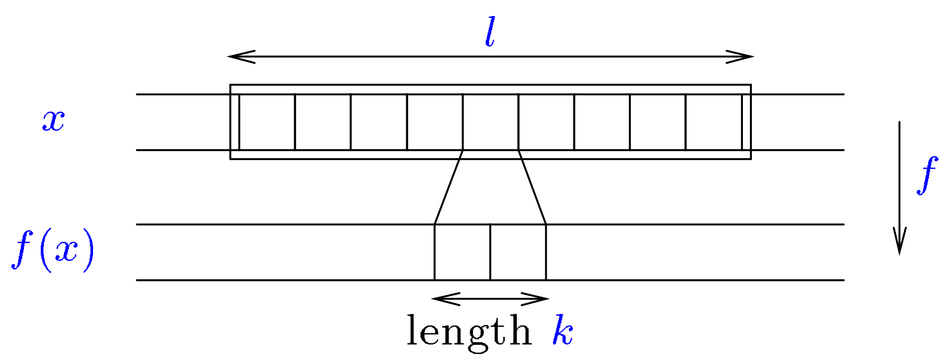


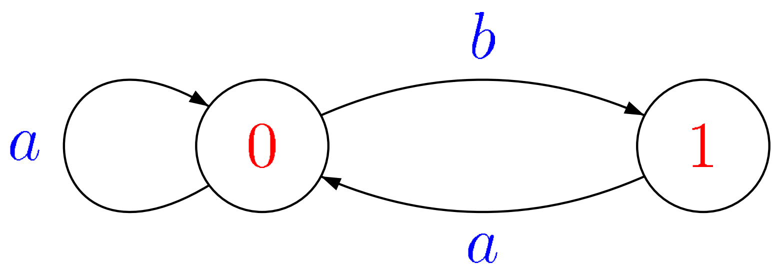
Figure 2: A k-synchronous function f

Automata and transducers

Automaton:

states · labels · transitions

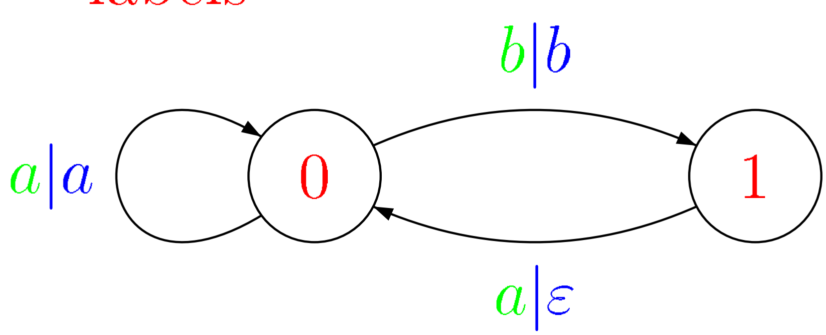
$$\mathcal{A} = (Q, A, E)$$



Transducer:

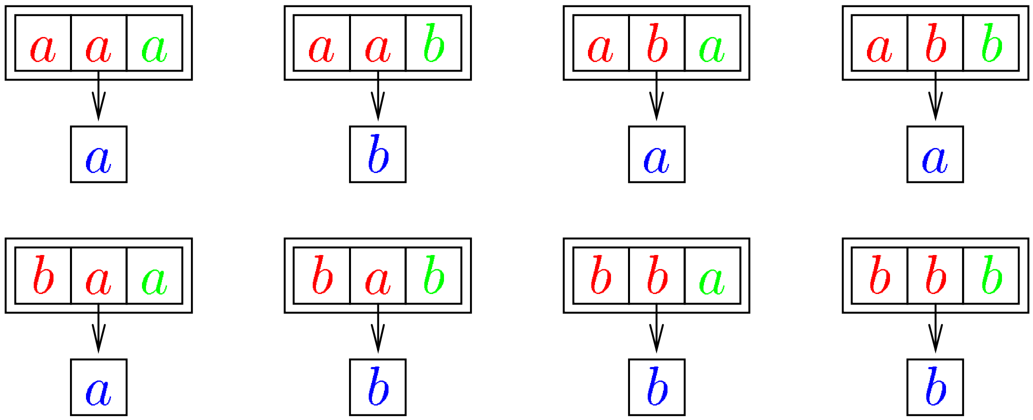
states · labels · transitions

$$\mathcal{T} = (Q, A \times B^*, E)$$

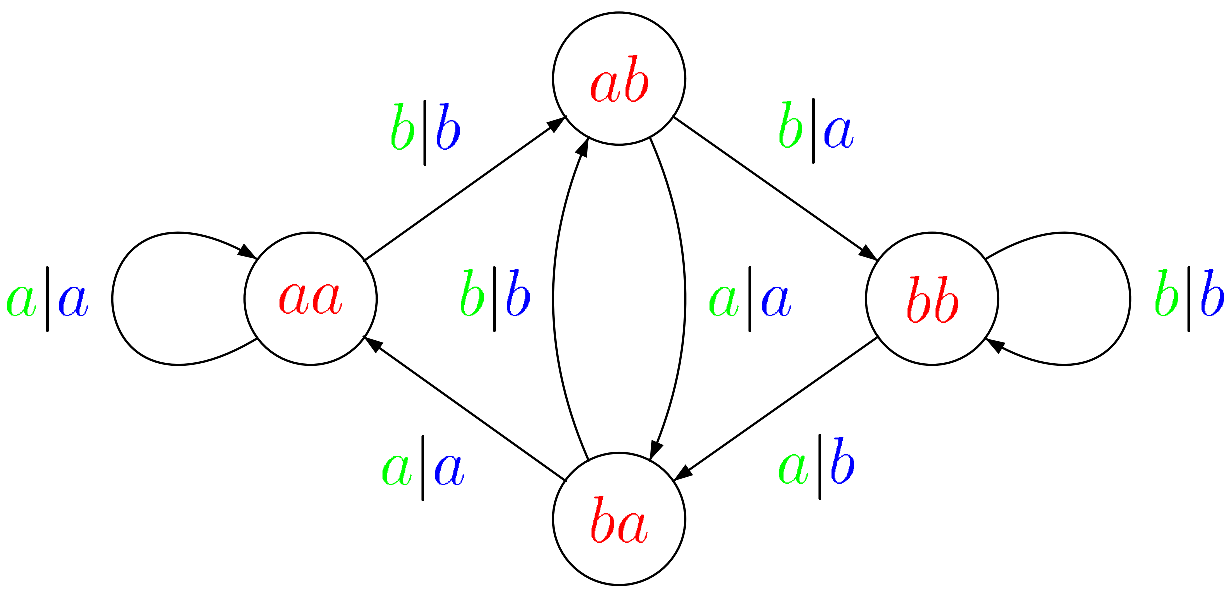


Functions and transducers

The sliding block map ($l = 3$)



replaces each factor ab by ba. It can be realized by the transducer



Local automata

In general, let $m + a = l - 1$ for

- m letters of memory
- a letters of anticipation.

$$\overbrace{\begin{matrix} a_1 a_2 \ldots a_m \\ a_{m+1} \ldots a_{l-1} \end{matrix}}^{\text{memory}}_{\underbrace{\hphantom{a_{m+1} \ldots a_{l-1}}}_{\text{anticipation}}} \xrightarrow{a_{m+1} | \bar{f}(a_1 \ldots a_l)} \overbrace{\begin{matrix} a_2 \ldots a_{m+1} \\ a_{m+2} \ldots a_l \end{matrix}}^{\text{memory}}_{\underbrace{\hphantom{a_{m+2} \ldots a_l}}_{\text{anticipation}}}$$

An automaton is (m, a)-local iff

$$p_0 \xrightarrow{a_1} p_1 \xrightarrow{a_2} \cdots \xrightarrow{a_{n+m}} p_{n+m}$$

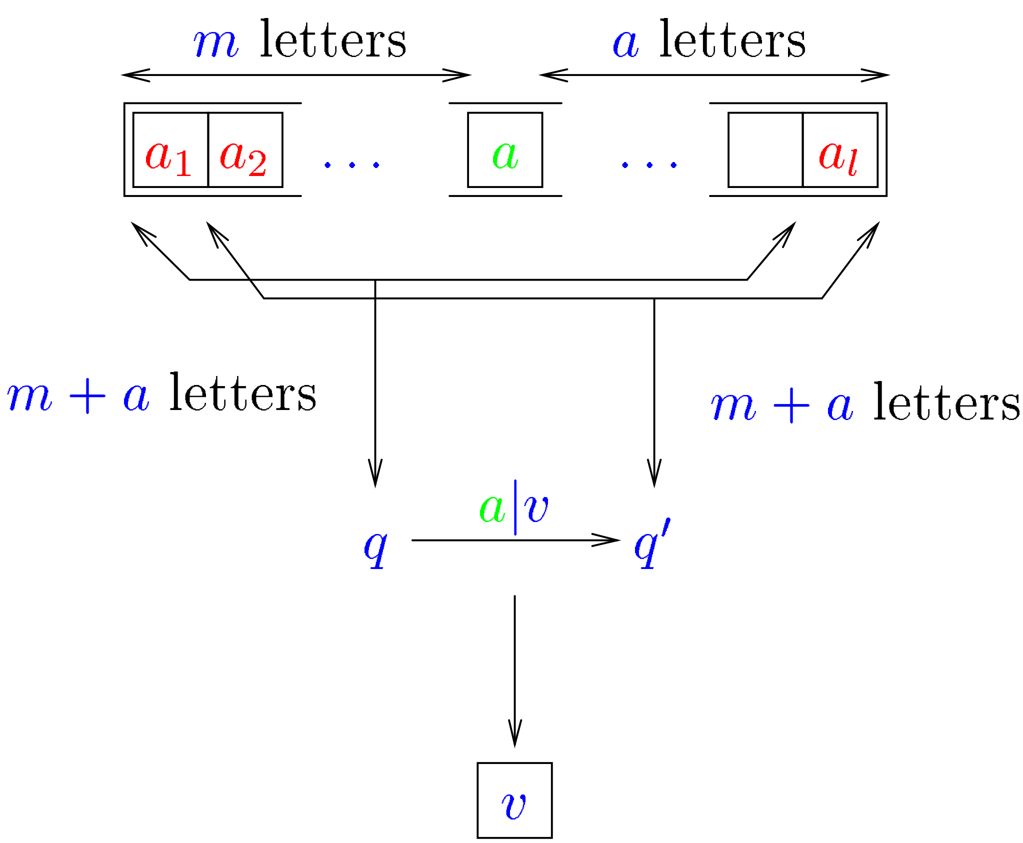
$$p'_0 \xrightarrow{a_1} p'_1 \xrightarrow{a_2} \cdots \xrightarrow{a_{n+m}} p'_{n+m}$$

then $p_m = p'_m$.

In a local automaton, any bi-infinite word labels at most one path.

Local automata and sliding blocks

Conversely, if f is realized by a (m, a)-local transducer, then, f is a sliding block map whose block size is $l = m + a + 1$.



Resynchronization

The problem:

- Input: sliding block map given by a strongly connected transducer.
- Goal: synchronization by keeping the locality of the transducer.
- Assumption: fixed transmission rate on cycling paths.

The transmission rate of a path

$$p \xrightarrow{u|v} q$$

is the quotient $|v|/|u|$

References

- Eilenberg and Schützenberger showed in [1] that a length preserving rational relation of $A^* \times B^*$ is a rational subset of $(A \times B)^*$. The proof is done on regular expressions.
- Frougny and Sakarovitch give in [2] an algorithm for synchronization of relations with bounded length difference. This constitutes another proof of the previous result. Their algorithm operates directly on the transducer that realizes the relation.

[1] Eilenberg, S. *Automata, Languages and Machines*, vol. A. Academic Press, New York, 1972.

[2] Frougny, C., and Sakarovitch, J. Synchronized relations of finite words. *Theoret. Comput. Sci. 108* (1993), 45–82.

The delays

Let k the transmission rate of the cycling paths.

A state i is distinguished and a delay $b(q)$ is associated to each state q.

- $b(i) = 0$;
- $b(q) = |v| - k|u|$ for any path $i \xrightarrow{u|v} q$.

A depth-first search verifies that the transmission rate is the same on all cycling paths and computes the delay $b(q)$ of each state.

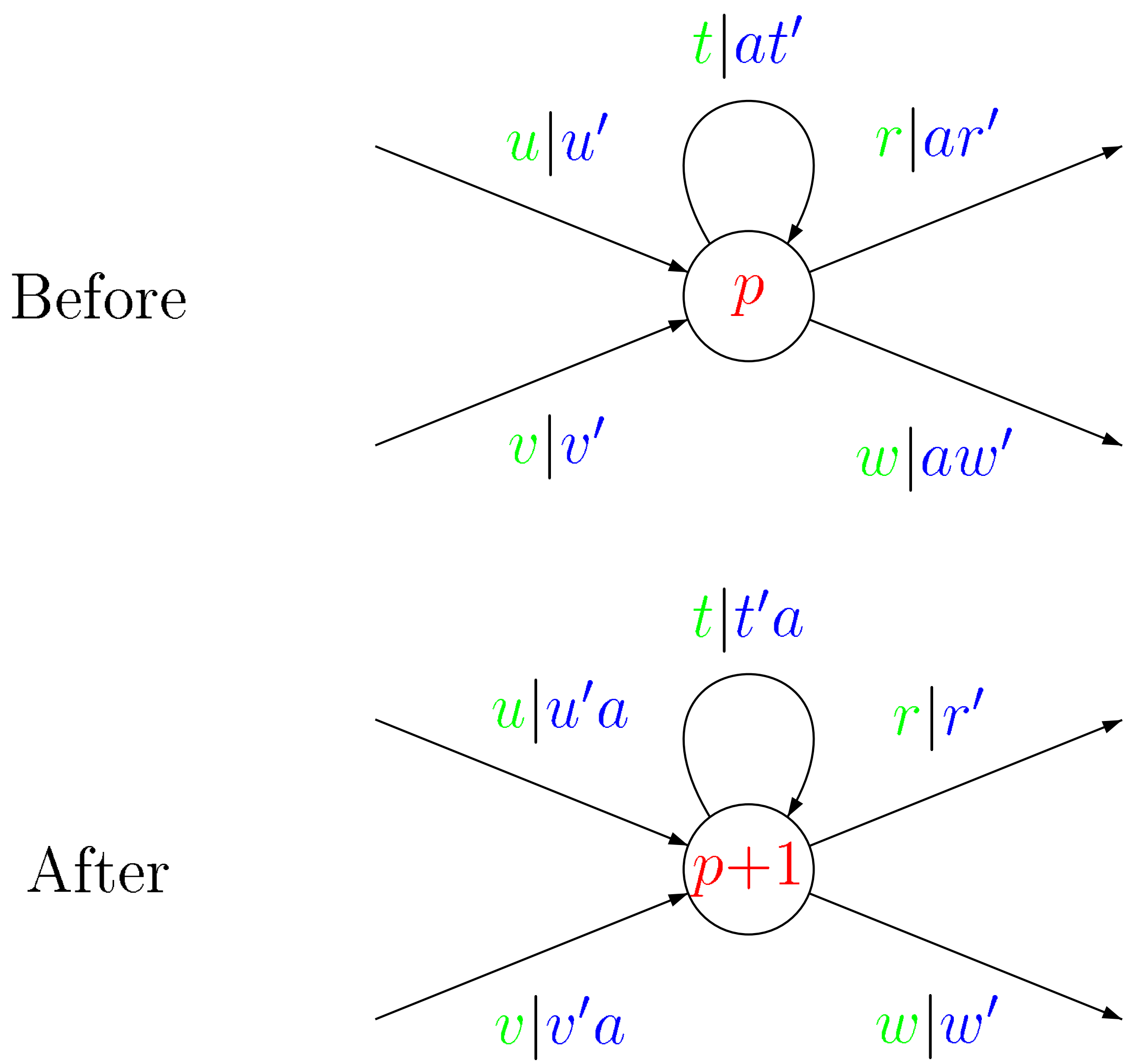
The delays are defined up to an additive constant and for any path, $q \xrightarrow{u|v} q'$,

$$b(q') - b(q) = |v| - k|u|$$

The transducer is synchronous iff $b(q) = 0$ for any state q.

Incrementation and decrementation

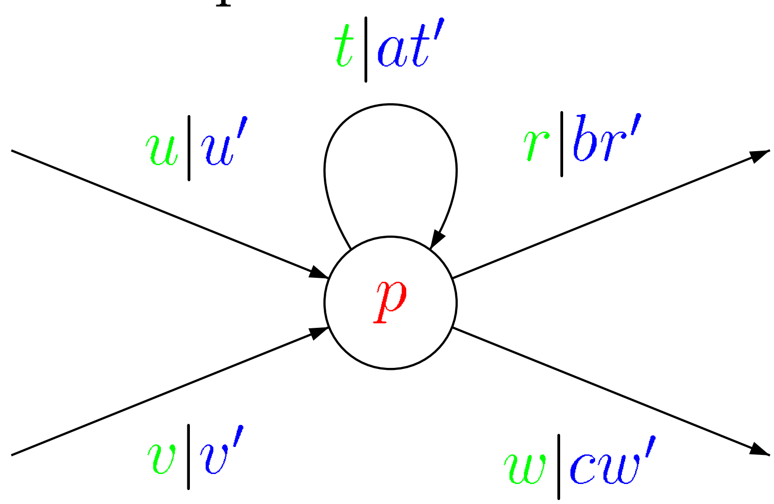
If the output labels of all outgoing transitions from a state q begin with the same letter a, it is possible to shift this letter to the incoming transitions.



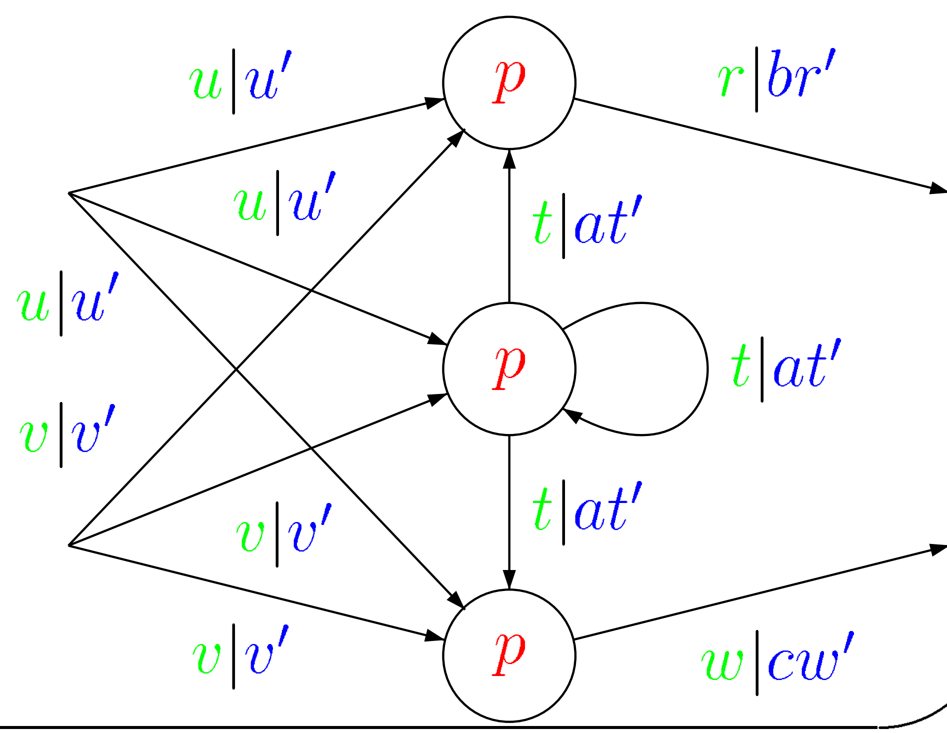
Output splitting

If the output labels of the outgoing transitions do not begin with the same letter, the state is split.

Before



After



The algorithm

The method consists in iterating the following process

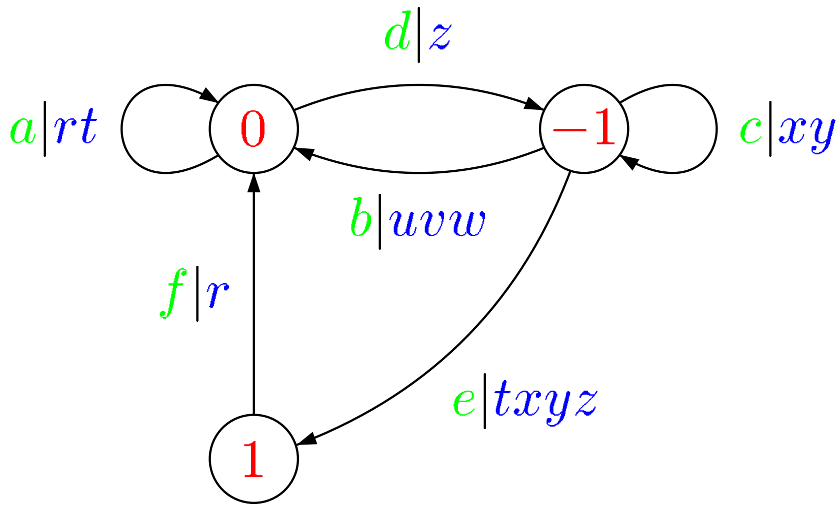
- Split and then increment all states having a negative delay which is maximal
- Split and then decrement all states having a positive delay which is maximal

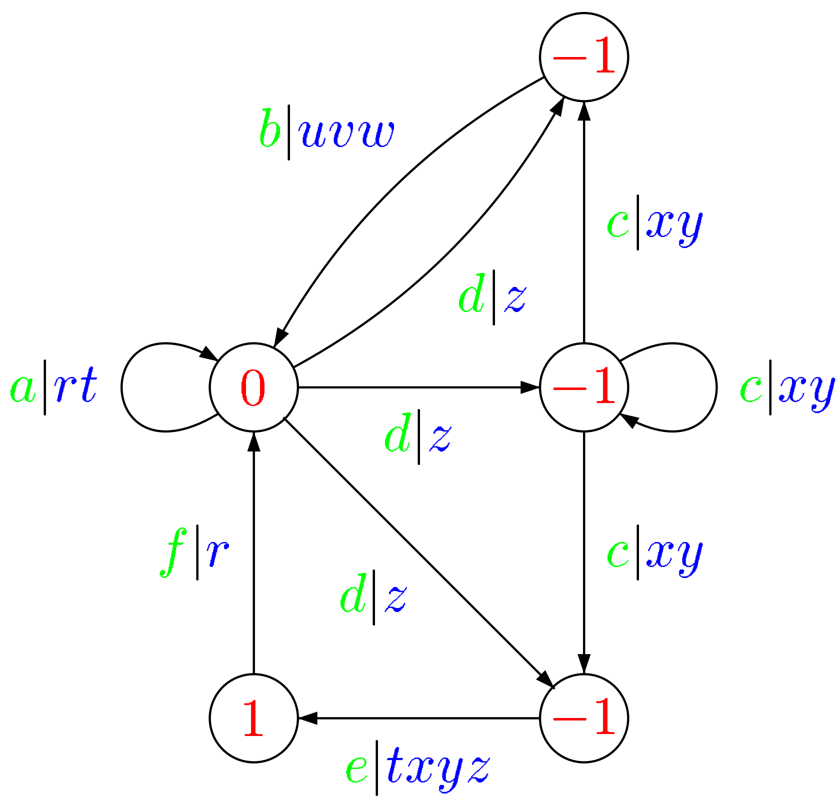
while there are non-zero delays.

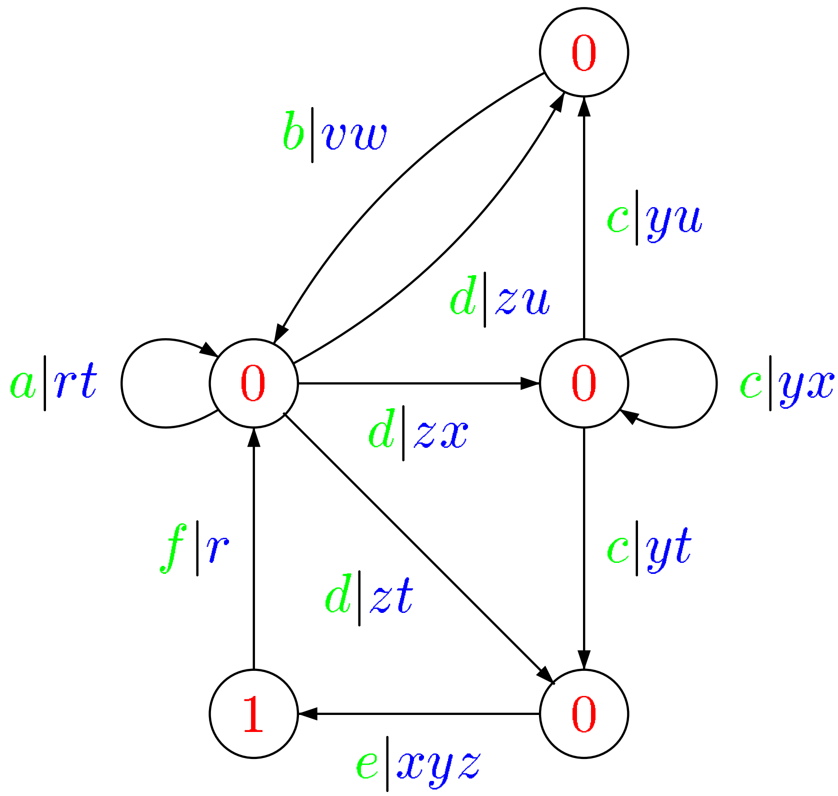
At each step, the maximal difference between delays decreases.

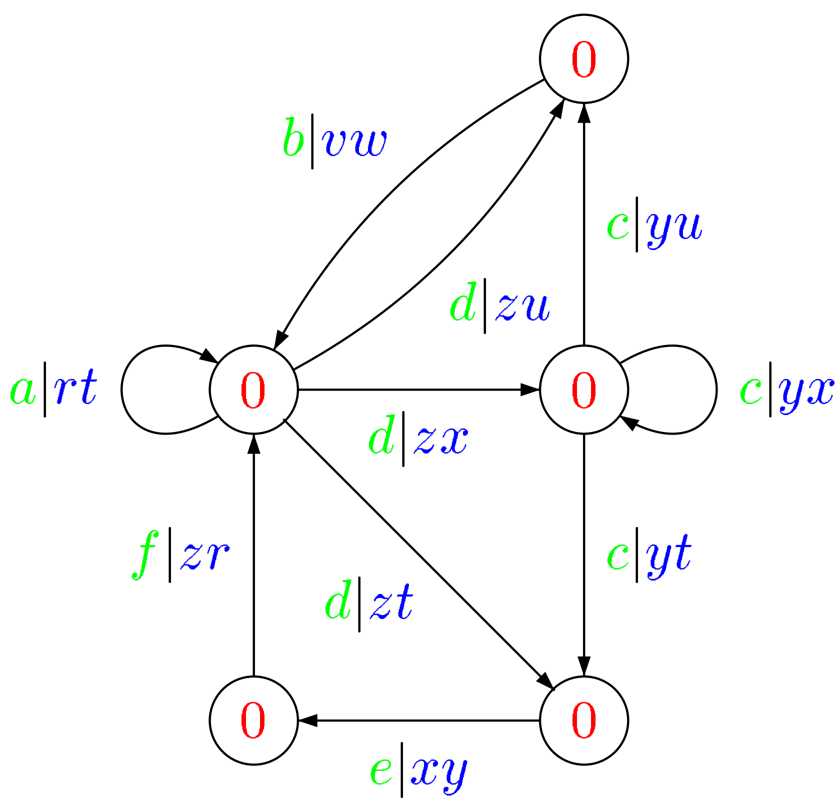
The number of iterations is thus bounded by the initial delays.

Example


d|z
a|rt
0
−1
c|xy
b|uvw
f|r
e|txyz
1


−1
b|uvw
c|xy
d|z
a|rt
0
−1
c|xy
d|z
f|r
d|z
c|xy
1
−1
e|txyz


0
b|vw
c|yu
d|zu
a|rt
0
0
c|yx
d|zx
f|r
c|yt
d|zt
1
0
e|xyz

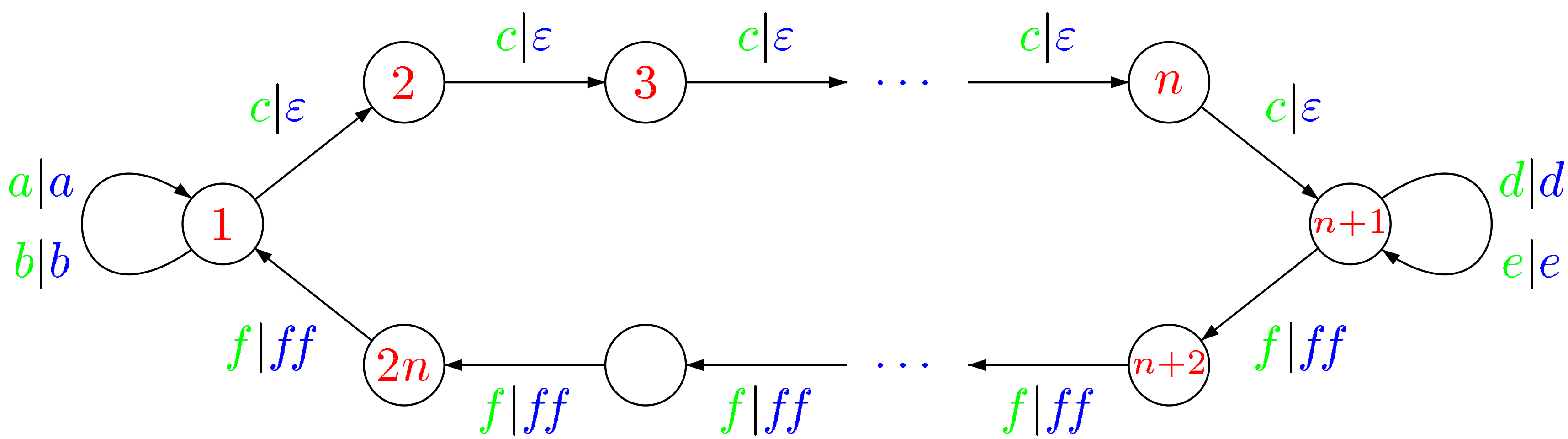

0
b|vw
c|yu
d|zu
a|rt
0
0
c|yx
d|zx
f|zr
c|yt
d|zt
0
0
e|xy

Complexity of the algorithm

The size of the sliding block is increased by 1 at each state splitting. The final size of the sliding block is bounded by the sum of the initial size and the maximal delay.

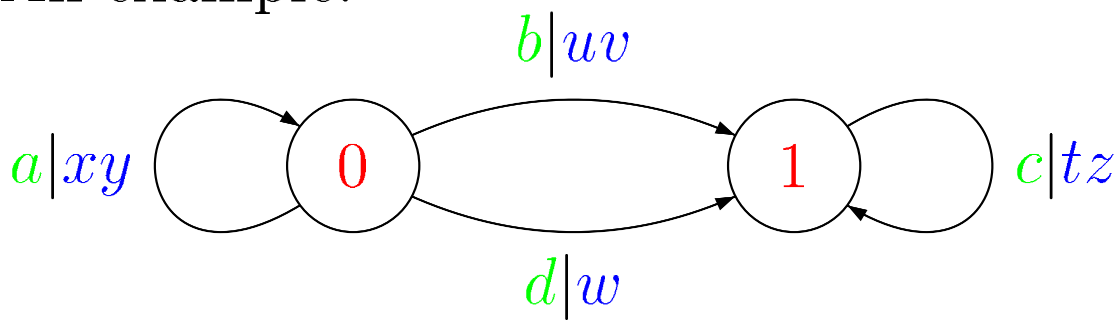
However, there might be an exponential blow up of the number of states. This is intrinsic to the synchronization process.

Proposition 1 *There are local transducers with n states, such that any synchronous local transducer which is equivalent has an exponential number of states.*

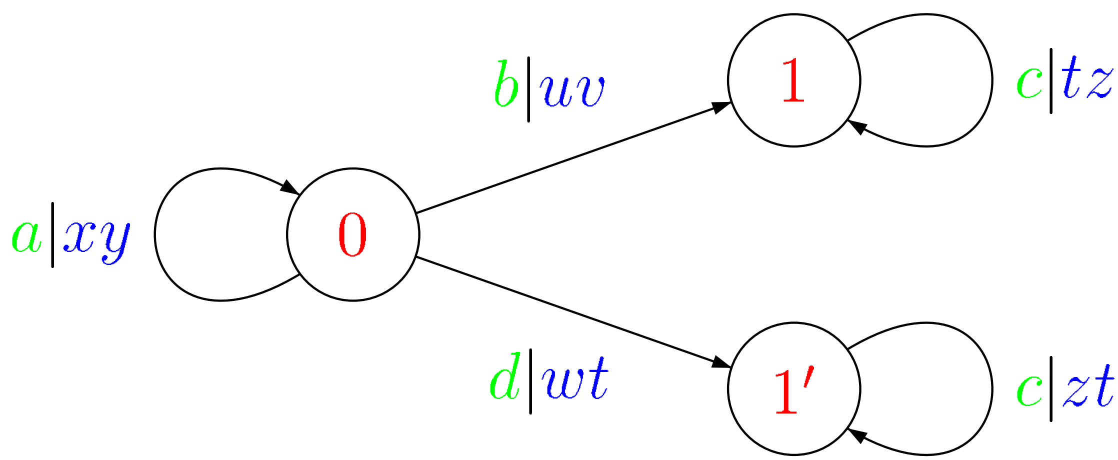

a|a
b|b
1
c|ε
2
c|ε
3
c|ε
…
c|ε
n
c|ε
n+1
d|d
e|e
f|ff
n+2
f|ff
…
f|ff
f|ff
2n
f|ff

Non strongly connected transducers

An example:



This transducer can be synchronized by duplicating the state 1.



The transducer is however not local.

Actually, this function cannot be realized by a local transducer.

Additional condition

Assume that any cycling path has a transmission rate equal to k.

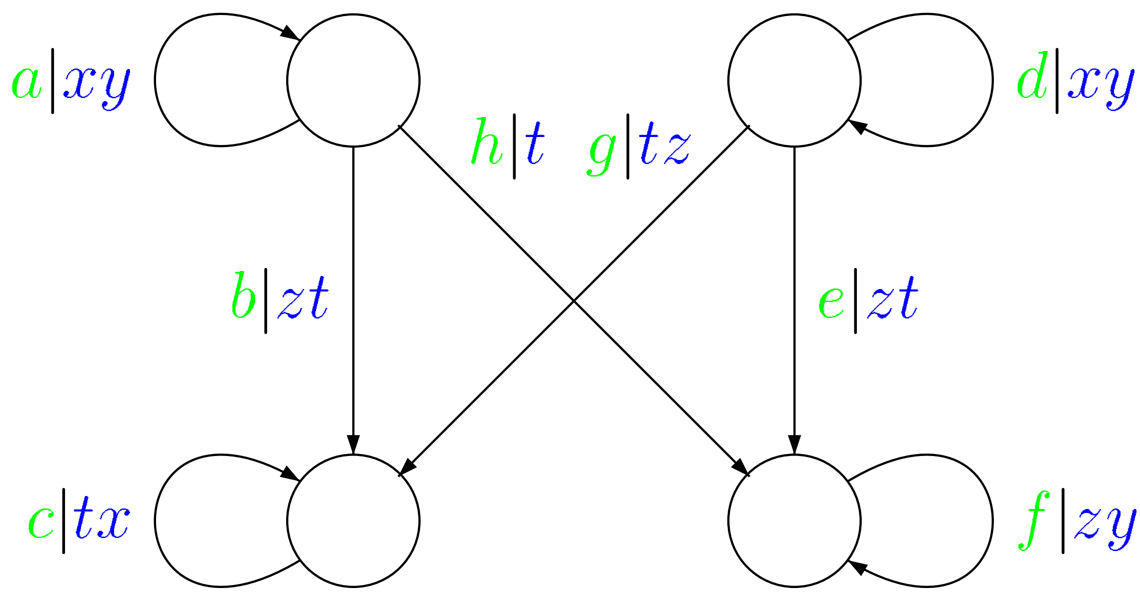
The transducer is then considered as an undirected graph. The valuation of a cycle is the sum over each transition of $|v| - k|u|$ if the direction of the transition is the same as the direction of the cycle and of the opposit $k|u| - |v|$ otherwise.

Condition : zero valuation for each cycle of the undirected graph.

Both conditions are equivalent if the graph is strongly connected.

Synchronization

This transducer has a transmission rate of 2 on any cycling path but it does not satisfy the condition.



Proposition 2 *Any transducer satisfying the above condition can be synchronized by state splitting.*