Workshop “Codes, Systems and Graphical Models”
Minneapolis, 1999

Deterministic Synchronization of Finite State Transducers

Christiane Frougny

Laboratoire Informatique Algorithmique: Fondements et Applications, Université Paris 7 and C.N.R.S.

Jacques Sakarovitch

Laboratoire Traitement et Communication de l’Information, C.N.R.S. and E.N.S.T., Paris.

Finite state transducers

Finite or infinite sequences on a finite alphabet.

Example: base β integer > 1, digit set
$A = \{0, \dots, \beta - 1\}$.
$A^* =$ Finite sequences $\longleftrightarrow$ positive integers.
$A^{\mathbb{N}} =$ Infinite sequences $\longleftrightarrow$ positive real numbers.

A *input* alphabet, B *output* alphabet.
Finite state transducer from A to B:
$\mathcal{T} = (Q, A^* \times B^*, E, I, \omega)$ is a directed labelled graph with set of edges E and set of states Q finite.
Labels of edges $\in A^* \times B^*$.
$I \subset Q =$ initial states.
$\omega : Q \longrightarrow B^*$ is the terminal function.

$\varphi : A^* \longrightarrow B^*$ is *computable* by $\mathcal{T}$ if its graph is

$$\{(f, g) \in A^* \times B^* \mid \exists i \in I, \text{ a path } i \stackrel{f/g'}{\to} q$$

$$\text{with } \omega(q) = g'',\ g = g'g''\}$$

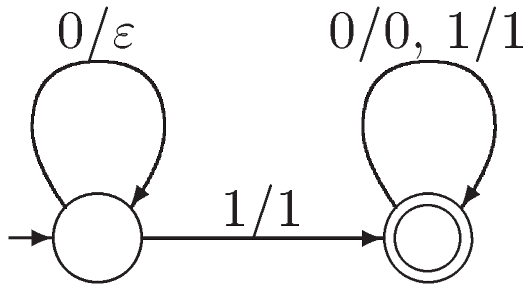
$\varphi : A^{\mathbb{N}} \longrightarrow B^{\mathbb{N}}$ is *computable* by $\mathcal{T}$ if its graph is the set of labels of infinite paths beginning in a state of I and going infinitely often through $T = \{q \in Q \mid \omega(q) \neq \emptyset\}$.

N.B. No terminal cycles u/ε ou ε/u

$\mathcal{T}$ is *input deterministic* (or *input right-resolving*) if

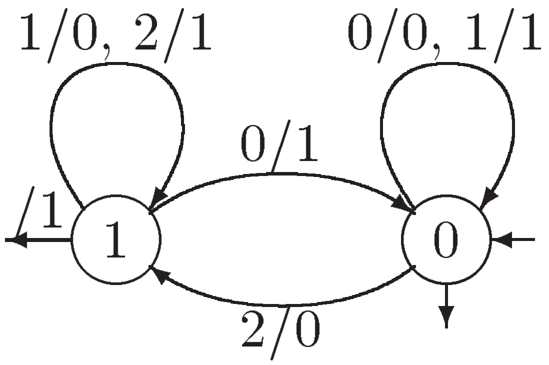
- label of edges $\in A \times B^*$
- $|I| = 1$
- $p \stackrel{a/u}{\rightarrow} q$ and $p \stackrel{a/u'}{\rightarrow} q'$ implies $q = q'$ and $u = u'$.

Example 1. The transducer which removes the 0's ahead.



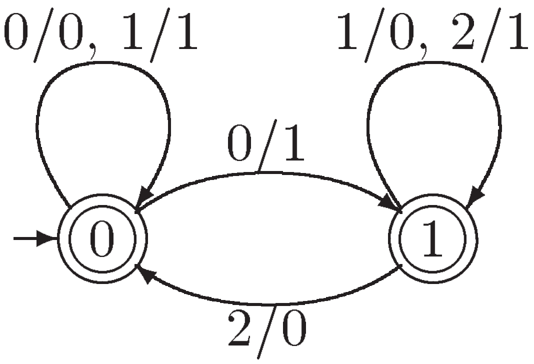
Example 2. Addition of integers base 2 (from right to left)

$$
\begin{array}{cccccc}
 & & 1 & 1 & 1 & 1 \\
+ & & 0 & 1 & 1 & 1 \\
\hline
 & & 1 & 2 & 2 & 2 \\
\hline\hline
 & 1 & 0 & 1 & 1 & 0
\end{array}
$$



This transducer is *input deterministic*, but from right to left, that is, Least Significant Digit First.

Example 3. Addition of real numbers in base 2 (from left to right)



This transducer is *not* input deterministic, but it is unambiguous. Useless for practical computations.

On-line arithmetic

To pipe-line additions/subtractions, multiplications and divisions, computations are to be done Most Sigificant Digit First, *i.e.* from left to right.

Similarly for the processing of real numbers (infinite expansions).

Additional requirement: digit serially processing after a certain delay δ of latency. To generate the jth digit of the result, it is necessary and sufficient to have the first $(j + \delta)$ digits of the input available.

[Ercegovac, 84]

On-line finite state transducer

On-line finite state transducer with delay δ:

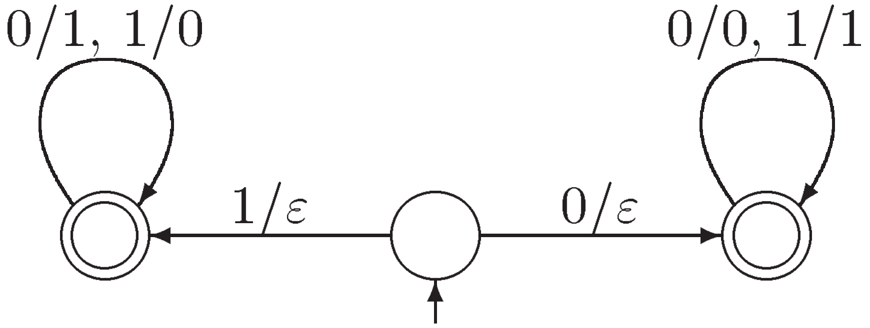
• input deterministic (from left to right)

• transient part: it reads without writing during a time δ

• synchronous part: after the delay, for one input letter there is one output letter.

Introduced by J.M. Muller [94].

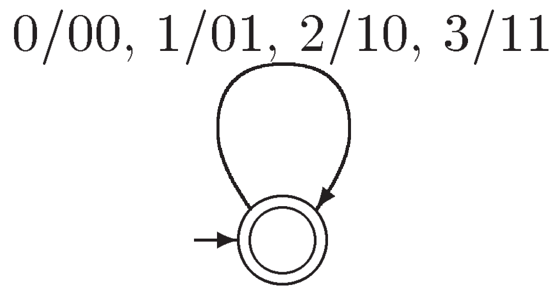
Example 4. On-line finite state transducer with delay 1, computes the tent function on $[0, 1]$ in base 2.



Finite state transducer with bounded rate

Finite state transducer with *bounded rate*: on every cycle, the lengths of the input and of the output are the same.

Example 5. This finite state transducer realizes the conversion from base 4 to base 2. It is input deterministic, but has no bounded rate.



This function is on-line computable with no delay using an infinite queue as an auxiliary memory.

THEOREM **1** . *Every input deterministic finite state transducer with bounded rate is equivalent to an on-line finite state transducer.*

Proof on an example from Computer Arithmetic.

Avizienis representations

Base β, digit set $D = \{\bar{a}, \ldots, a\}$ with $\beta/2 \leq a \leq \beta - 1$. Redundancy.

Example 6. $\beta = 2$, $D = \{\bar{1}, 0, 1\}$. Addition with no propagation of the carry [CR, 78].

$$
\begin{array}{cccccc}
 & & 1 & 0 & 1 & 1 \\
+ & & 0 & 1 & 1 & 1 \\
\hline
 & & 1 & 1 & 2 & 2 \\
\hline
 & 1 & 1 & 1 & 1 & \\
 & & \bar{1} & \bar{1} & 0 & 0 \\
\hline\hline
 & 1 & 0 & 0 & 1 & 0
\end{array}
$$

Rewriting rules:

$$2 \quad \rightarrow \quad {}^{1}\big|_{0}$$

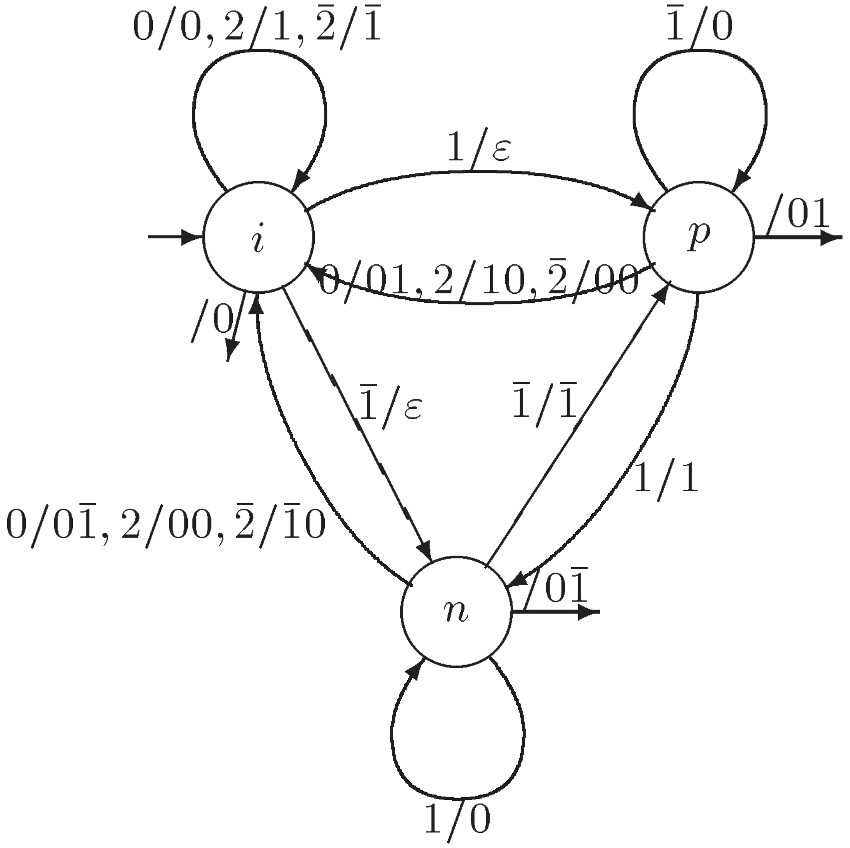
For digit 1, one uses a window:

$$1 \quad \rightarrow \quad {}^{1}\big|_{\bar{1}} \qquad \text{if the digit right to 1 is} \geq 0$$

otherwise $1 \quad \rightarrow \quad {}^{0}\big|_{1}$

Similarly for negative digits.

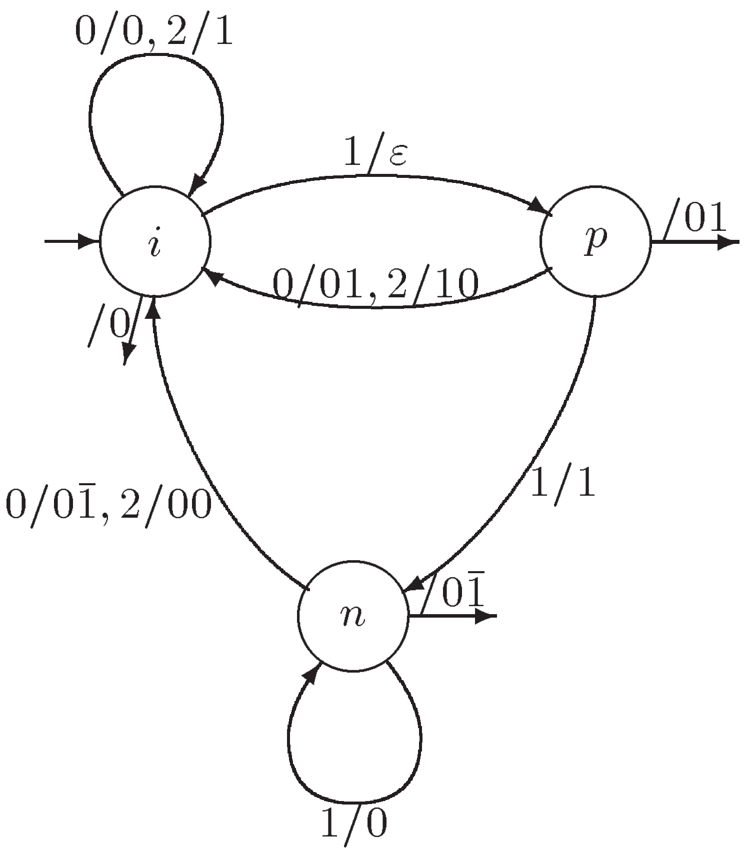
Example 7. Input deterministic transducer realizing addition in base 2 on $\{\bar{1}, 0, 1\}$



Converts a word $e_k e_{k-1} \cdots$ with $e_i \in \{\bar{2}, \ldots, 2\}$ into a word $d_{k+1} d_k \cdots$ with $d_i \in \{\bar{1}, 0, 1\}$ such that

$$\sum_{i \le k} e_i 2^i = \sum_{i \le k+1} d_i 2^i$$

Example 8. Transducer $\mathcal{D}_0$, the same restricted to inputs on $\{0, 1, 2\}$.



ALGORITHM:

Input: $\mathcal{D} = (Q, A \times B^*, E, i, \omega)$ input deterministic transducer with bounded rate.

Output:

$\mathcal{L} = (R = R_s \cup R_t, A \times (B \cup \varepsilon), F, (i, \varepsilon), \psi)$ on-line transducer equivalent to $\mathcal{D}$.

Step 1. It is assumed that for each state $q \in Q$, for every path $i \overset{f/g}{\rightarrow} q$, the value $|f| - |g|$ is constant, denoted by $\rho(q)$, the delay in state q.

Ex: $\rho(i) = 0$, $\rho(p) = 1$ and $\rho(n) = 1$

Step 2. For each state $q \in Q$ let $\delta(q)$ be the *greatest variation of the delay on every path starting in* q

$$\delta(q) = \max\{\rho(t) - \rho(q) \mid \ t \text{ accessible from } q\ \}$$

Ex: $\delta(i) = \max\{0, 1\} = 1$,
$\delta(p) = \delta(n) = \max\{-1, 0\} = 0$

Step 3. Construction of the synchronous part $\mathcal{L}_s$.

- States $R_s = \{(q,u) \in Q \times B^* \mid |u| = \delta(q)\}$.

  Ex: $R_s = \{(i,0),(i,1),(i,\bar{1}),(p,\varepsilon),(n,\varepsilon)\}$

- Synchronous edges of $\mathcal{L}_s$:

$$(q,u) \stackrel{a/b}{\to} (t,v) \in \mathcal{L}_s \Leftrightarrow q \stackrel{a/f}{\to} t \in \mathcal{D} \text{ and } bv = uf$$

- Terminal function $\psi((q,u)) = u\omega(q)$.

Step 4. Construction of the transient part $\mathcal{L}_t$.

- States $R_t = \{(q,u) \in Q \times B^* \mid |u| < \delta(q)\}$.

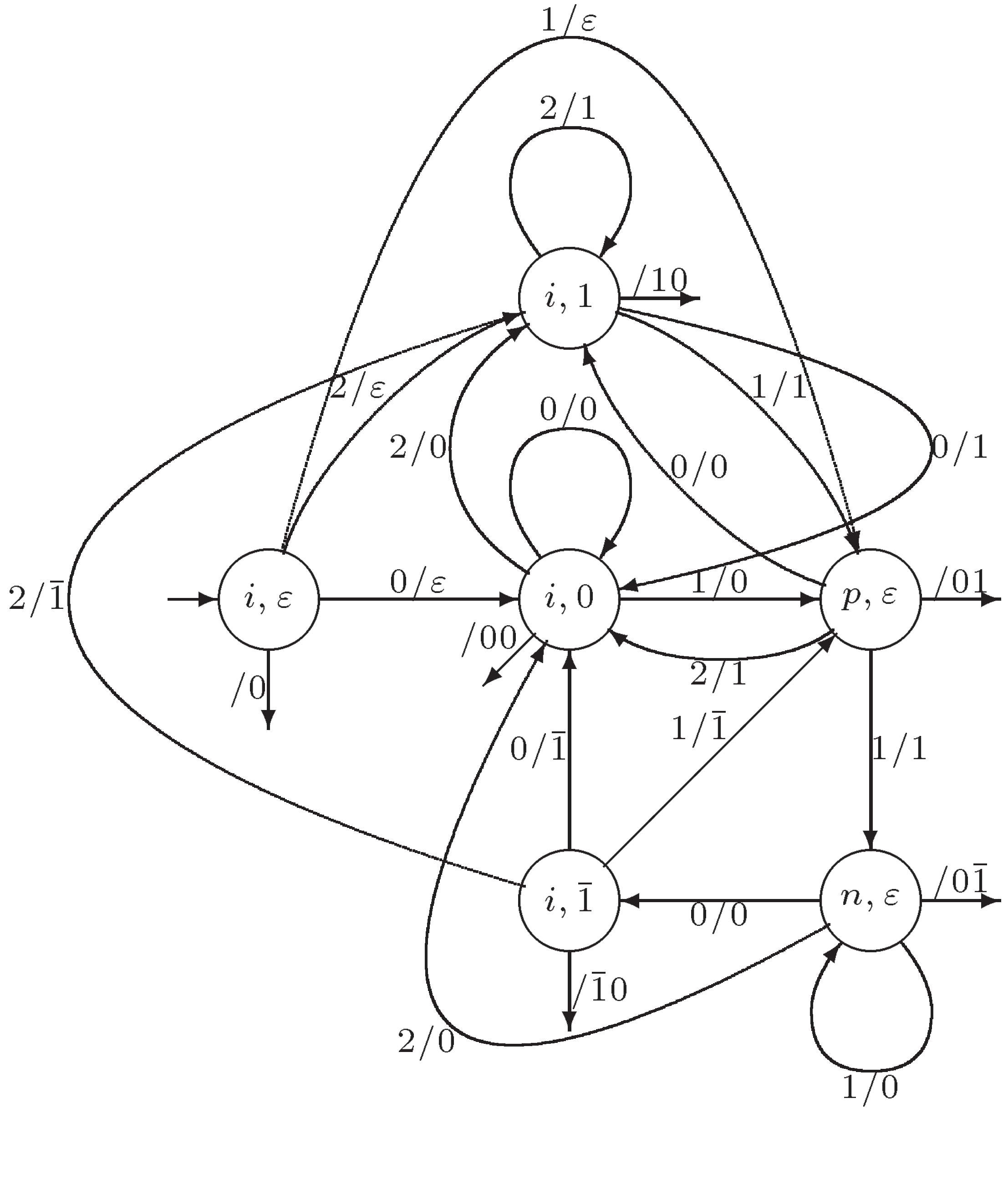
  Ex: $R_t = \{(i,\varepsilon)\}$

- Transient edges

$$(q,u) \stackrel{a/\varepsilon}{\to} (t,uf) \in \mathcal{L}_s \Leftrightarrow q \stackrel{a/f}{\to} t \in \mathcal{D}$$

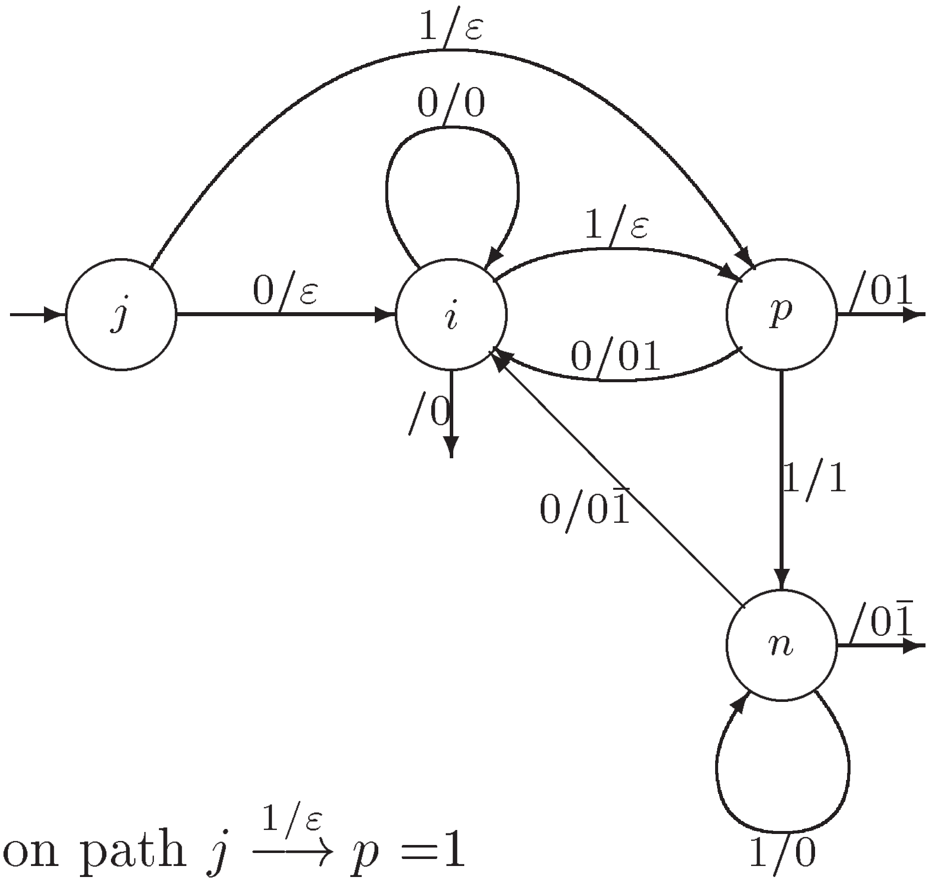
- Terminal function $\psi((q,u)) = u\omega(q)$.

Example 8. On-line transducer $\mathcal{L}_0$ with delay 1, equivalent to $\mathcal{D}_0$.



Return on **Step 1**. It is not always possible to compute the function ρ.

Example 9. Transducer $\mathcal{D}_1$, almost the same as $\mathcal{D}_0$ restricted to $\{0, 1\}$.



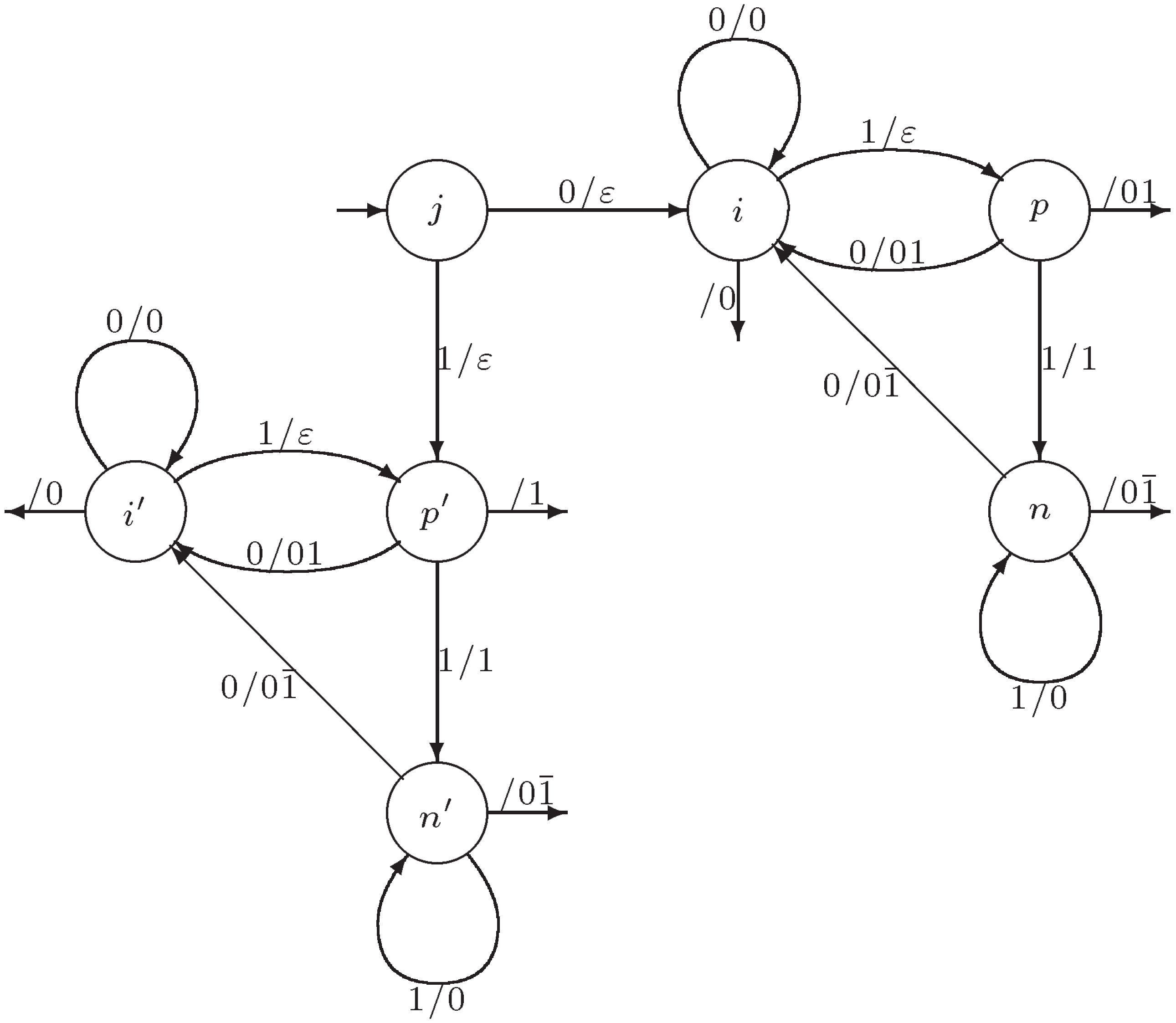
Delay on path $j \xrightarrow{1/\varepsilon} p$ =1

Delay on path $j \xrightarrow{0/\varepsilon} i \xrightarrow{1/\varepsilon} p = 2$.

Construction of a transducer $\mathcal{E}$ equivalent to $\mathcal{D}$ such that the delay can be computed for each state [FS, 93].

Step 0. Each state q of $\mathcal{D}$ is splitted in all the states (q, z) where z is a relative integer.
Each transition $q \xrightarrow{f/g} t$ of $\mathcal{D}$ is lifted up in all the transitions $(q, z) \xrightarrow{f/g} (t, z + |f| - |g|)$ in $\mathcal{E}$.
The initial state of $\mathcal{E}$ is $(i, 0)$ for i initial state of $\mathcal{D}$.
The trimmed part of $\mathcal{E}$ is finite since $\mathcal{D}$ has a bounded rate.

Example 9. Transducer $\mathcal{E}_1$ equivalent to $\mathcal{D}_1$.
$j = (j, 0)$, $i = (i, 1)$, $i' = (i, 0)$, $p = (p, 2)$,
$p' = (p, 1)$, $n = (n, 2)$, $n' = (n, 1)$.



$\rho(j) = 0$, $\rho(i) = 1$, $\rho(p) = 2$, $\rho(n) = 2$, $\rho(p') = 1$, $\rho(i') = 0$, $\rho(n') = 1$, $\delta(j) = 2$, $\delta(i) = 1$, $\delta(p) = 0$, $\delta(n) = 0$, $\delta(p') = 0$, $\delta(i') = 1$, $\delta(n') = 0$.

The definition of the synchronous part relies on the property that, for each transition $q \xrightarrow{a/f} t$ of $\mathcal{E}$, we should get $\delta(t) = \delta(q) + |f| - |a|$ but it is not true for transition $j \xrightarrow{1/\varepsilon} p'$.

Let

$$\Delta = \max\{\rho(q) \mid q \in Q\}\,.$$

For every q in Q let

$$\zeta(q) = \Delta - \rho(q)\,.$$

Then for each $q \xrightarrow{f/g} t$ of $\mathcal{E}$

$$\zeta(t) = \zeta(q) + |g| - |f|$$

thus, $\delta(q) \leq \zeta(q)$.

If q and t are in the same strongly connected component of $\mathcal{E}$ then $\zeta(q) - \delta(q) = \zeta(t) - \delta(t)$.

Ex: $\Delta = 2$, $\zeta(j) = 2$, $\zeta(i) = 1$, $\zeta(p) = 0$, $\zeta(n) = 0$, $\zeta(p') = 1$, $\zeta(i') = 2$, $\zeta(n') = 1$.

ζ allows the construction of the on-line transducer $\mathcal{L}$ equivalent to $\mathcal{E}$ instead of the function δ in step 3 and 4.

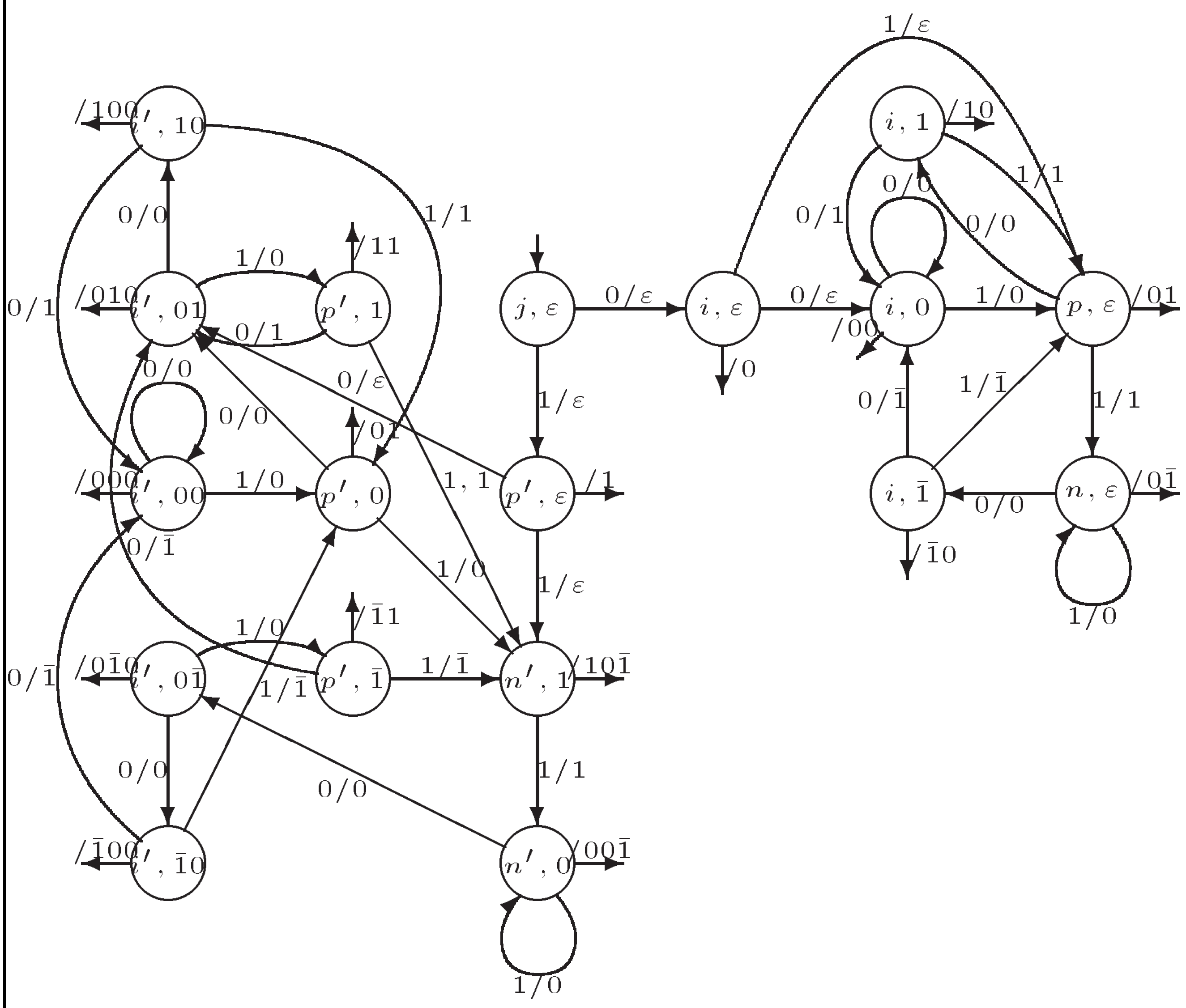
It only remains to check that the delay on each path from the initial state (j, ε) of $\mathcal{D}$ arriving in a state (q, u) of $\mathcal{D}_s$ is equal to

$$\rho(q) + |u| = \rho(q) + \zeta(q) = \rho(q) + (\Delta - \rho(q)) = \Delta\,.$$

Thus the automaton $\mathcal{L}$ has a uniform delay, equal to Δ.

The resulting transducer has a size proportional to $|Q| \times |B|^{\Delta}$. The complexity of the algorithm is proportional to the size of the resulting transducer.

Example 9. On-line transducer with delay 2 equivalent to $\mathcal{E}_1$.



References

[1] A. Avizienis, Signed-digit number representations for fast parallel arithmetic. *IRE Transactions on electronic computers* **10** (1961), 389–400.

[2] C.Y. Chow and J.E. Robertson, Logical design of a redundant binary adder. *Proc. 4th Symposium on Computer Arithmetic* (1978), 109–115.

[3] M.D. Ercegovac, On-line arithmetic: An overview. *Real time Signal Processing VII* **SPIE 495** (1984), 86–93.

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

[5] Ch. Frougny et J. Sakarovitch, Synchronisation déterministe des automates à délai borné. *Theoret. Comput. Sci.* **191** (1998), 61–77.

[6] J.-M. Muller, Some characterizations of functions computable in on-line arithmetic. *I.E.E.E. Trans. on Computers* **43** (1994), 752–755.