Preservation of normality by transducers

Olivier Carton[1] Elisa Orduna[2]

[1]IRIF
Université Paris Diderot & CNRS

[2]Universidad de Buenos Aires
partially funded by LIA INFINIS

ANR Delta
December 2018

Context

Random sequence $\rightsquigarrow$ Transformation $\rightsquigarrow$ Random sequence ?

Context

Normal sequence $\rightsquigarrow$ Transducer $\rightsquigarrow$ Normal sequence ?

Outline

Context

Normal words

Deterministic transducers

Selectors

Results

Weighted automata

A weighted automaton for frequencies

Deciding preservation of normality

Normal words

A normal word is an infinite word such that all finite words of the same length occur in it with the same frequency.
More precisely, let A be an alphabet.

Definition

If $x \in A^\omega$ and $w \in A^*$, the frequency of w in x is defined as follows:

$$\mathrm{freq}(x, w) = \lim_{n \to \infty} \frac{|x[1 \ldots n]|_w}{n}$$

where $|z|_w$ denotes the number of occurrences of w in z.

A word $x \in A^\omega$ is normal if for each $w \in A^*$:

$$\mathrm{freq}(x, w) = \frac{1}{|A|^{|w|}}$$

Normal Words (continued)

Theorem (Borel, 1909)

The decimal expansion of almost every real number in $[0, 1)$ *is a normal word in the alphabet* $\{0, 1, ..., 9\}$.

Nevertheless, not so many examples have been proved normal. Some of them are:

- Champernowne 1933 (natural numbers):

  $$123456789101112131415161718192021222324 25\cdots$$

- Besicovitch 1935 (squares):

  $$149162536496481100121144169196225256289324\cdots$$

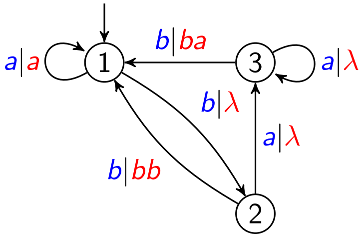
- Copeland and Erdős 1946 (primes):

  $$235711131719232931374143475359616771737983\cdots$$

Transducers

An input-deterministic transducer (aka sequential) is a deterministic automaton whose transitions, not only consume a symbol from an input alphabet A, but also produce a finite word in the output alphabet B as output.

Example



Preservation of normality

A functional transducer $\mathcal{T}$ is said to preserve normality if for every normal word $x \in A^\omega$, $\mathcal{T}(x)$ is also normal.

Question

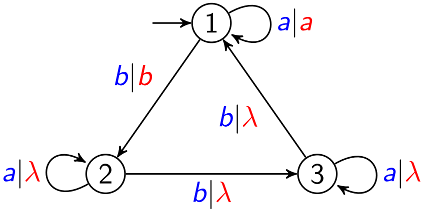
Given a deterministic complete transducer $\mathcal{T}$, does $\mathcal{T}$ preserve normality?

Selectors

A selector is a complete input-deterministic transducer such that:

- each transition is either of type $p \xrightarrow{a|a} q$ or of type $p \xrightarrow{a|\lambda} q$.
- all transitions starting from each state p have the same type.

Example

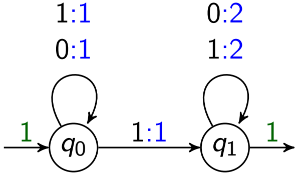


Theorem (Agafonov 68)

Selectors do preserve normality.

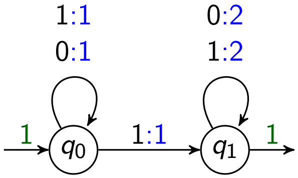
Weighted Automata

A weighted automaton $\mathcal{T}$ is an automaton whose transitions, not only consume a symbol from an input alphabet A, but also have a transition weight in $\mathbb{R}$ and whose states have initial weight and final weight in $\mathbb{R}$.



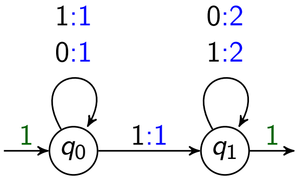
This weighted automaton computes the value of a binary number.

The weight of a run $q_0 \xrightarrow{b_1} q_1 \xrightarrow{b_2} \cdots \xrightarrow{b_n} q_n$ in $\mathcal{A}$ is the product of the weights of its n transitions times the initial weight of q_0 and the final weight of q_n.



$$\text{weight}_{\mathcal{A}}(q_0 \xrightarrow{1} q_0 \xrightarrow{1} q_1 \xrightarrow{0} q_2) = 1 * 1 * 1 * 2 * 1 = 2$$

The weight of a run $q_0 \xrightarrow{b_1} q_1 \xrightarrow{b_2} \cdots \xrightarrow{b_n} q_n$ in $\mathcal{A}$ is the product of the weights of its n transitions times the initial weight of q_0 and the final weight of q_n.



The weight of a word w in $\mathcal{A}$ is given by the sum of weights of all runs labeled with w:

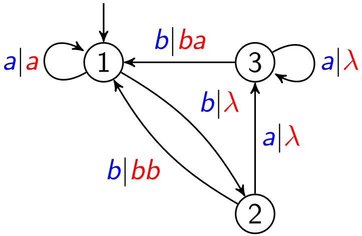
$$\mathrm{weight}_{\mathcal{A}}(w) = \sum_{\gamma \text{ run on } w} \mathrm{weight}_{\mathcal{A}}(\gamma)$$

$$\begin{aligned} \mathrm{weight}_{\mathcal{A}}(110) \; = \; & \mathrm{weight}_{\mathcal{A}}(q_0 \xrightarrow{1} q_0 \xrightarrow{1} q_1 \xrightarrow{0} q_1) + \\ & \mathrm{weight}_{\mathcal{A}}(q_0 \xrightarrow{1} q_1 \xrightarrow{1} q_1 \xrightarrow{0} q_1) \quad = 2 + 4 = 6 \end{aligned}$$

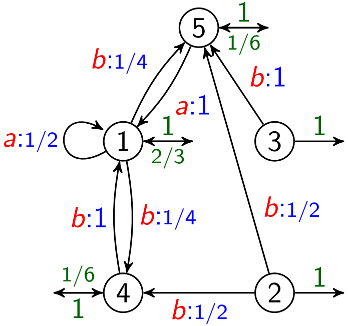
Theorem

For every strongly connected deterministic transducer $\mathcal{T}$ there exists a weighted automaton $\mathcal{A}$ such that for any finite word w and any normal word x, $\text{weight}_{\mathcal{A}}(w)$ is exactly the frequency of w in $\mathcal{T}(x)$.

Example



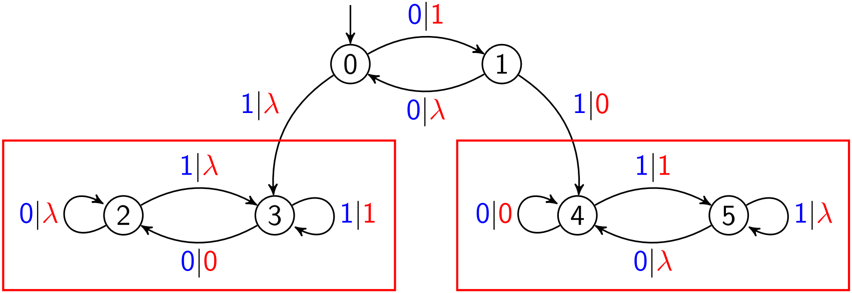
Transducer $\mathcal{T}$



Weighted Automaton $\mathcal{A}$

Recurrent strongly connected components

A strongly connected component is recurrent if it has no outgoing transitions.



Key ingredients

Fact 1

A run labeled with a normal word always reaches a recurrent SCC.

Key ingredients

Fact 1

A run labeled with a normal word always reaches a recurrent SCC.

Fact 2

Each state of a SCC transducer has a frequency in a run labeled by a normal word.
This frequency is given exactly by the stationary distribution of the weighted automaton interpreted as a Markov chain.
Hence, the frequency with which each state of a SCC transducer is visited is the same for any normal word.

Fact 3

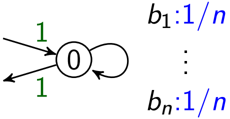
From any state q of a SCC transducer, all paths of the same length starting at q are visited with the same frequency when consuming a normal input word.

Deciding preservation of normality

Proposition

Such a weighted automaton can be computed in cubic time with respect to the size of the transducer.

To determine whether $\mathcal{T}$ preserves normality, the automaton $\mathcal{A}$ can be compared to the automaton $\mathcal{B}$ that realizes the expected frequencies $1/|A|^{|w|}$ for any finite word.



The comparison between these automata can be made using Schützenberger's algorithm, and it is decidable as all weights are rational numbers.

Future work

- Enlarge the class of transducers for which the algorithm solves the problem.
- Adapt the algorithm to solve similar problems.

Algorithm

Input: A deterministic complete transducer $\mathcal{T}$.
Output: `True` if $\mathcal{T}$ preserves normality, `False` otherwise.

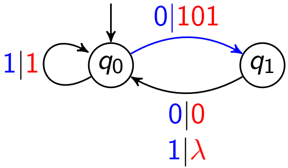
- For each recurrent strongly connected component S of $\mathcal{T}$:
  - Build a weighted automaton associated to $\mathcal{T}$.
    - Normalize the transducer
    - Build a weighted automaton $\mathcal{A}$ using the normalized transducer
    - Assign $\mathcal{A}$'s states final and initial weights.
  - Using $\mathcal{A}$ analyze whether $\mathcal{T}$ preserves normality.

We use Kosaraju's algorithm to find the set of strongly connected components of $\mathcal{T}$ and then filter the ones that are recurrent.

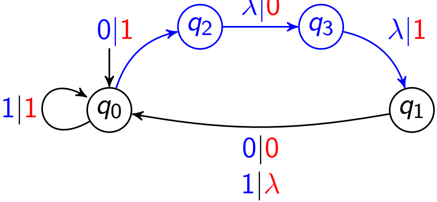
Normalizing the transducer

We normalize the transducer $\mathcal{T}$ so that the output of any transition has length at most

Example



Original



Normalized

Construction of the weighted automaton

Motivation

We aim to calculate $\mathrm{freq}(\mathcal{T}(x), w)$ for any normal word $x \in A^\omega$ and any word $w \in B^*$.

We first solve the this auxiliary problem:

Compute the frequency in the infinite run of each finite sequence of transitions of the form

$$p \xrightarrow{a_1|\lambda} q_1 \xrightarrow{a_2|\lambda} q_2 \cdots q_{n-1} \xrightarrow{a_n|\lambda} q_n \xrightarrow{a_{n+1}|b} q$$

for each pair of states p, q and for each $b \in B$.

Construction of the weighted automaton

- The states of $\mathcal{A}$ are the same as in $\mathcal{T}$
- For each pair of states p, q, and each symbol $b \in B$, there is a transition $p \xrightarrow{b} q$.
- The weight $\mathrm{weight}(p \xrightarrow{b} q)$ of a transition is precisely the frequency of finite sequences of transitions from p to q that produce exactly b

Construction of the weighted automaton

Procedure

1. Assigns weight to the transducer's transitions:
   - transitions with non empty input have weight $1/|A|$,
   - transitions with empty input have weight 1,

Construction of the weighted automaton

Procedure

2. Consider the matrix E whose (p, q) entry has the sum of the weights of transitions of the form $p \xrightarrow{a|\lambda} q$.
3. Compute $E^* = Id + E + E^2 + \cdots + E^n + \cdots$. Note that:
   - The matrix E^n has in its (p, q) entry the frequency with which paths of length n from p to q with output λ are taken:

     $$p \xrightarrow{a_1|\lambda} q_1 \xrightarrow{a_2|\lambda} q_2 \cdots q_{n-1} \xrightarrow{a_n|\lambda} q$$

   - The matrix E^* has in its (p, q) entry the frequency with which paths from p to q of any length with output λ are taken.

Construction of the weighted automaton

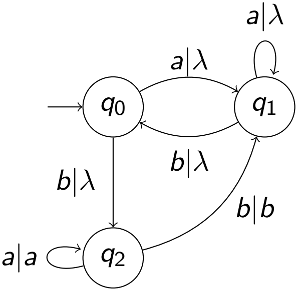
Procedure

4. For each $b \in B$, consider the matrix N_b having in its (p, q) entry the sum of the weights of transitions of the form $p \xrightarrow{a|b} q$.
5. Define the weighted automaton $\mathcal{A}$ so that

$$\text{weight}(p \xrightarrow{b} q) = (E^* \cdot N_b)_{p,q}$$

Construction of the weighted automaton

Example



Transducer $\mathcal{T}$

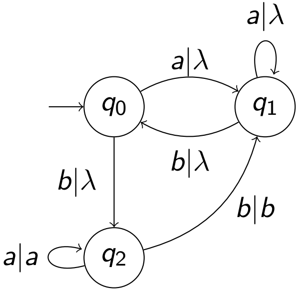
$$E = \begin{bmatrix} 0 & 1/2 & 1/2 \\ 1/2 & 1/2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad E^* = \begin{bmatrix} 2 & 2 & 1 \\ 2 & 4 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$E^*.N_a = \begin{bmatrix} 0 & 0 & 1/2 \\ 0 & 0 & 1/2 \\ 0 & 0 & 1/2 \end{bmatrix}$$

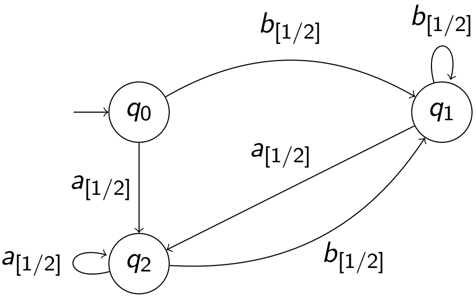
$$E^*.N_b = \begin{bmatrix} 0 & 1/2 & 0 \\ 0 & 1/2 & 0 \\ 0 & 1/2 & 0 \end{bmatrix}$$

Construction of the weighted automaton

Example



Transducer $\mathcal{T}$



Automaton $\mathcal{A}$

Assign initial and final weights to states

Procedimiento

i. Consider the matrix T that in its (p, q) entry has the sum of the weights of the transitions $p \xrightarrow{b_{[w]}} q$ of $\mathcal{A}$.

   The matrix T is an stochastic matrix, and has an associated stochastic distribution, in other words, a vector π such that:

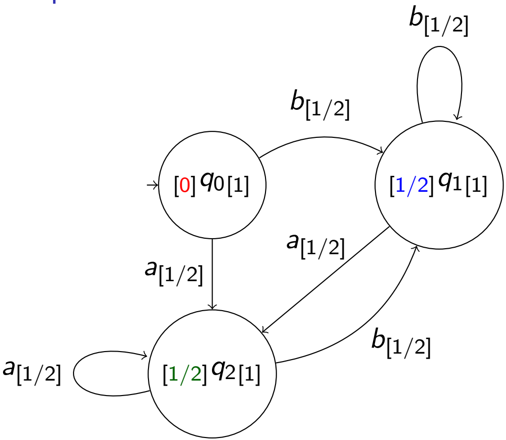
$$\pi \cdot T = \pi$$

ii. Assign the i-th state initial weight π_i.

iii. Assign every state final weight 1.

Assign initial and final weights to states

Example



$$T = \begin{bmatrix} 0 & 1/2 & 1/2 \\ 0 & 1/2 & 1/2 \\ 0 & 1/2 & 1/2 \end{bmatrix}$$

$$\pi = \begin{bmatrix} 0 & 1/2 & 1/2 \end{bmatrix}$$