Probabilistic and Frequency Finite-State Transducers *

Kaspars Balodis, Anda Beriņa, Gleb Borovitsky, Rūsiņš Freivalds, Ginta Garkāje, Vladimirs Kacs, Jānis Kalējs, Iļja Kucevalovs, Jānis Ročāns, Madars Virza

Faculty of Computing, University of Latvia, Raiņa bulvāris 29, Riga, LV-1459, Latvia

Abstract. A transducer is a finite-state automaton with an input and an output. We compare possibilities of nondeterministic and probabilistic transducers, and prove several theorems which establish an infinite hierarchy of relations computed by these transducers. We consider only left-total relations (where for each input value there is exactly one allowed output value) and Las Vegas probabilistic transducers (for which the probability of any false answer is 0). It may seem that such limitations allow determinization of these transducers. Nonetheless, quite the opposite is proved; we show a relation which can only be computed by probabilistic (but not deterministic) transducers, and one that can only be computed by nondeterministic (but not probabilistic) transducers.
Frequency computation was introduced by Rose and McNaughton in early sixties and developed by Trakhtenbrot, Kinber, Degtev, Wechsung, Hinrichs and others. It turns out that for transducers there is an infinite hierarchy of relations computable by frequency transducers and this hierarchy differs very much from similar hierarchies for frequency computation by a) Turing machines, b) polynomial time Turing machines, c) finite state acceptors.

1 Introduction

Frequency computation was introduced by G. Rose [13] as an attempt to have a deterministic mechanism with properties similar to probabilistic algorithms. The definition was as follows. A function f: $w \to w$ is (m, n)-computable, where $1 \leq m \leq n$, iff there exists a recursive function R: $w^n \to w^n$ such that, for all n-tuples $(x_1, \cdots, x_n)$ of distinct natural numbers,

$$card\{i : (R(x_1, \cdots, x_n))_i = f(x_i)\} \geq m.$$

R. McNaughton cites in his survey [12] a problem (posed by J. Myhill) whether f has to be recursive if m is close to n. This problem was answered by B.A. Trakhtenbrot [14] by showing that f is recursive whenever $2m > n$. On

* The research was supported by Grant No. 09.1570 from the Latvian Council of Science and by Project 2009/0216/1DP/1.1.1.2.0/09/IPIA/VIA/044 from the European Social Fund.

the other hand, in [14] it was proved that with $2m = n$ nonrecursive functions can be (m, n)-computed. E.B. Kinber extended the research by considering frequency enumeration of sets [9]. The class of (m, n)-computable sets equals the class of recursive sets if and only if $2m > n$.

For resource bounded computations, frequency computability behaves differently. For example, it is known that whenever $n' - m' > n - m$, under any reasonable resource bound there are sets which are (m', n')-computable, but not (m, n)-computable. However, for finite automata, an analogue of Trakhtenbrot's result holds: the class of languages (m, n)-recognizable by deterministic finite automata equals the class of regular languages if and only if $2m > n$. Conversely, for $2m > n$, the class of languages (m, n)-recognizable by deterministic finite automata is uncountable for a two-letter alphabet [1]. When restricted to a one-letter alphabet, every (m, n)-recognizable language is regular. This was also shown by Kinber.

Frequency computations became increasingly popular when the relation between frequency computation and computation with a small number of queries was discovered [11, 7, 2, 3].

We considered problems similar to those in the classical papers [14, 9, 1] for finite-state transducers. We found the situation to be significantly different.

A finite state transducer is a finite state machine with two tapes: an input tape and an output tape. These tapes are one-way, i.e. the automaton never returns to the symbols once read or written. Transducers compute relations between the input words and output words. A deterministic transducer produces exactly one output word for every input word processed.

In this paper we consider advantages and disadvantages of nondeterministic, deterministic, frequency and probabilistic transducers. Obviously, if a relation is such that several output words are possible for the same input word, then the relation cannot be computed by a deterministic transducer. For this reason, in this paper we restrict ourselves to relations which produce exactly one output word for every input word processed.

Definition 1. *We say that a relation $R(x, y)$ is* left-total*, if for arbitrary x there is exactly one y satisfying $R(x, y)$.*

Probabilistic algorithms may be of several types: those allowing errors of all types, Monte Carlo, Las Vegas, etc. Since our relations produce exactly one output word for every input word processed, it was natural to consider only Las Vegas transducers, i.e. probabilistic transducers for which the probability of every false answer is 0. It follows immediately from the definition that every relation computed by such a probabilistic transducer can be computed by a nondeterministic transducer as well. We prove below that nondeterministic transducers are strictly more powerful than Las Vegas transducers.

The zero probability of all possible false results of a probabilistic transducer has another unexpected consequence. It turns out that only recursive relations of small computational complexity can be computed by probabilistic transducers with a probability, say, $\frac{1}{4}$ or $\frac{1}{7}$. Hence a natural question arises, whether every

rational number $0 \leq p \leq 1$ can be the best probability to compute some relation by a probabilistic finite-state transducer. It turns out that it is not the case. We show examples of relations that can be computed by probabilistic finite-state transducers with probabilities $\frac{1}{2}$, $\frac{1}{3}$, $\frac{1}{4}$, etc. and not better. We believe that no other best probabilities can exist for relations of the type considered by us.

This hierarchy of relations turns out to be related to the number of symbols of help needed for deterministic finite-state transducers that take advice to compute these relations.

2 Definitions

We use standard definitions of deterministic, nondeterministic and probabilistic transducers, which are well-established in theoretical computer science literature [5]. Our model is slightly different in the regard that we allow multiple output symbols for each transition, however it can be easily seen that the expressive power of transducer is unaffected.

Definition 2. *A nondeterministic transducer D is a six-tuple $\langle Q, \Sigma, \Delta, \delta, q_0, F \rangle$, which satisfies the following conditions:*

- *Q is a finite set, whose members are called states of D. $q_0 \in Q$ is called the initial state of D,*
- *Σ is a set of input symbols of D and is called the input alphabet,*
- *Δ is a set of output symbols of D and is called the output alphabet,*
- *δ is a relation from $Q \times (\Sigma \cup \{\epsilon\})$ to $Q \times \Delta^*$. Each tuple $((q, \alpha), (p, \beta))$ is called a transition rule of D,*
- *$F \subseteq Q$ is the set of accepting or final states of D*

A transducer operates in discrete time $t = 1, 2, 3 \ldots$ and handles two one-way tapes, where one is read-only input tape and contains a string from the input alphabet Σ and second is write-only output tape with finite output alphabet Δ.

The computation begins at the start state q_0. The computation from state q proceeds by reading symbol α from the input tape, following a suitable transition rule $((q, \alpha), (p, \beta))$ (the new state being p) and writing the corresponding output β to the output tape. The only exception is so called ϵ-transition $((p, \epsilon), (q, \beta))$, where the transducer changes the state and possibly writes an output without reading an input symbol.

We consider all our transducers as machines working infinetely long. At every moment, let x be the word having been read from the input tape up to this moment, and let y be the word written on the output tape up to this moment. Then we say that the pair (x, y) belongs to the relation computed by the transducer.

Definition 3. *A deterministic transducer is a nondeterministic transducer such that transition relation δ is a function from $Q \times \Sigma$ to $Q \times \Delta^*$, i.e., for each input state and each input symbol the corresponding output state and output symbols are uniquely determined according to the transition table δ and there are no ϵ-transitions.*

Definition 4. *A probabilistic transducer is a transducer of the form* $\langle Q, \Sigma, \Delta, \Psi, q_0, F\rangle$ *where the transition function* Ψ *is probabilistic, i.e., for every state* q *and input symbol* α*, output state* p *and output symbol* β *there is an associated probability with which transition occurs. And if* Ψ *contains an* ϵ*-transition from a state* q *then there is no transition from this state that reads an input symbol. We furthermore stipulate that probabilities of all transitions from any fixed pair of input state and input symbol (or* ϵ*) must sum to one.*

Definition 5. *We say that a probabilistic transducer* A *computes the relation* R *with probability* p *if for arbitrary pair* $(x, y) \in R$ *when* A *has read input* x *the probability to be in an accepting state having produced the output* y *is no less than* p*.*

Definition 6. *We say that a probabilistic transducer* A *is a Las Vegas transducer computing the relation* R *with probability* p *if for arbitrary pair* $(x, y) \in R$ *when* A *has read input* x *the probability to be in an accepting state having produced the output* y *is no less than* p*, and the probability to be in an accepting state having produced output other than* y *equals* 0*.*

We now introduce a new model of transducers, which is modeled after automata that take advice [6]. We call this model "transducers with help".

Definition 7. *A transducer with* n *help symbols is a deterministic transducer that receives on a special input a single symbol* α *of extra information such that* $\alpha \in \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$*, which is always accessible, regardless of the current position on the input tape.*

We say that a pair (x, y) *is in the relation computed by the transducer if there is an* $\alpha \in \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ *such that the transducer transforms the input* x *and the help-symbol* α *into output* y*.*

Since we consider in our paper only left-total relations and since our probabilistic transducers are Las Vegas, it easily follows that the transducer for arbitrary x and for arbitrary $\alpha \in \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ either produces a correct output y or reaches a non-accepting state.

Definition 8. *A frequency transducer with parameters* (m, n) *is a deterministic transducer with* n *input tapes and* n *output tapes. Every state of the transducer is defined as* $(a_1, a_2, \cdots, a_n)$*-accepting where each* $a_i \in \{$ *accepting* , *nonaccepting* $\}$*. We say that a left-total relation* R *is* (m, n)*-computed by the transducer if for arbitrary* n*-tuple of pairwise distinct input words* $(x_1, x_2, \cdots, x_n)$ *there exist at least* m *distinct values of* x_i *such that the* i*-th output* y_i *satisfies* $(x_i, y_i) \in R$*.*

Please notice that we do not demand the state of the frequency transducer to be all-accepting after the reading of the input words. This allows us introduce a counterpart of Las Vegas transducer.

Definition 9. *A Las Vegas frequency transducer with parameters* (m, n) *is a deterministic transducer with* n *input tapes and* n *output tapes. Every state of the*

transducer is defined as $(a_1, a_2, \cdots, a_n)$*-accepting where each* $a_i \in \{$ *accepting* , *nonaccepting* $\}$. *We say that a left-total relation* R *is* (m, n)*-computed by the transducer if for arbitrary* n*-tuple of pairwise distinct input words* $(x_1, x_2, \cdots, x_n)$*: 1) there exist at least* m *distinct values of* x_i *such that the* i*-th output* y_i *satisfies* $(x_i, y_i) \in R$*, and 2) if and only if a result* y_i *is produced on any output tape* i *and the current state of the transducer is* $(a_1, a_2, \cdots, a_{i-1}, accepting, a_{i+1}, \cdots, a_n)$*-accepting, then* $R(x_i, y_i)$.

We consider frequency transducers like all other transducers as machines working infinitely long. At every moment, let x be the word having been read from the input tape up to this moment, and let y be the word written (and accepted) on the output tape up to this moment. Then we say that the pair (x, y) belongs to the relation computed by the transducer. However, in the case when the input alphabet contains less than n letters, there is a problem how to define (m, n)-computation correctly.

Definition 10. *We say that a frequency transducer is performing a* strong (m, n)*-computation of a relation* R *if* at moments when all the input words $(x_1, x_2, \cdots, x_n)$ are distinct *there exist at least* m *distinct values of* x_i *such that the* i*-th output* y_i *satisfies* $(x_i, y_i) \in R$.

Definition 11. *We say that a frequency transducer is performing a* weak (m, n)*-computation of a relation* R *with parameters* $(b_1, b_2, \cdots b_n)$*, where* $b_1, b_2, \cdots b_n$ *are distinct integers, if the transducer is constructed in such a way that in the beginning of the work the transducer reads the first* b_1 *symbols from the input word* x_1*, the first* b_2 *symbols from the input word* b_2*,* $\cdots$ *, the first* b_n *symbols from the input word* x_n *and at all subsequent moments reads exactly 1 new symbol from every input tape. This ensures that at all moments the input words* $(x_1, x_2, \cdots, x_n)$ *are distinct. There is no requirement of the correctness of the results when the length of input words is* $(b_1, b_2, \cdots b_n)$ *but at all moments afterwards there exist at least* m *distinct values of* x_i *such that the* i*-th output* y_i *satisfies* $(x_i, y_i) \in R$.

3 Probabilistic transducers

First, we show that in our setting if a relation can be computed probabilistically with a probability greater than $\frac{1}{2}$ then it can be computed deterministically.

Theorem 1. *If a left-total relation* R *can be computed by a Las Vegas transducer with probability greater than* $\frac{1}{2}$ *then this relation can also be computed by a deterministic transducer.*

Proof. Our reduction will be in two steps. We will first show how to create a nondeterministic transducer that computes the same relation and then determinize it.

Let us take the Las Vegas transducer P that computes the relation R with probability greater than $\frac{1}{2}$ and "drop" the probabilities, i.e. create a nondeterministic transducer N that has the same set of states as the probabilistic

transducer and for every transition of the Las Vegas transducer, which has been associated with a non-zero probability, define a corresponding transition in the nondeterministic transducer. The initial state and set of accepting states are copied directly.

Note that for any input word a state of N is reachable if and only if the corresponding state of P is reachable with a non-zero probability. Also note that the probability of false results is zero (if an accepting state can be reached with a positive probability, then the corresponding output is indeed correct). This proves that pair (x, y) is a part of the relation computed by P if and only if (x, y) is a part of the relation computed by N, because a path with non-zero probability in P corresponds to a path in N and vice versa.

We can now use a powerset construction to determinize N. This demands an elaboration because not every nondeterministic transducer can be determinized.

We will create our deterministic transducer D as follows. D will have "metastates" that will correspond to subsets of the states of N; we will build metastates of D incrementally. A metastate will be accepting iff the corresponding subset of N's states contains an accepting state.

For n_0, the initial state of N, define $\{n_0\}$ to be the initial metastate of D. Mark this initial metastate as unvisited.

We now claim a following lemma: denote by A_i the set of accepting states that N can be in after reading i symbols of a fixed input word w. Then for any input word w and any i there is a transition from a state from A_i to a state from A_{i+1}.

Proof: We will prove this lemma by contradiction. Assume the contrary that all transitions to the states of A_{i+1} are not from A_i. Then the total probability of being in states from A_{i+1} after reading $i + 1$ symbols can be no greater than the probability of not being in states from A_i after reading i symbols. However, the latter probability is less than $\frac{1}{2}$ (as it is the complement of probability in being in states from A_i after reading i symbols), but the former probability must be greater than $\frac{1}{2}$. This gives the desired contradiction.

While there is an unvisited metastate $S = \{s_1, \ldots, s_k\}$ do the following:

- compute all possible metastates $\{t_1, \ldots, t_l\}$, such that each t_i can be reached from one of $s_j \in S$ by the means of one transition of N. Add these metastates to D and mark them as unvisited.
- for each new metastate T, if S and T are both accepting then add to D a transition from S to T (any transition from N transforming one of the accepting states of S to an accepting state of T). We claim that such transitions are well-defined. This stems from the fact that R is functional, false positive probability being zero and the lemma.
- mark S as visited.

We can now see that any computational path of N that leads to an accepting state can also be modeled by a computational path of D that leads to an accepting state and vice versa, therefore D is equivalent to N. By the construction of N it means that D is equivalent to P.

□

We believe to an even stronger result that the only interesting acceptance probabilities are $\frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots$ i.e. if a relation can be computed with a probability between them then it can also be computed with a higher probability. However we have not been able to prove this yet.

Conjecture 1. For every natural n if a left-total relation R can be computed by a Las Vegas transducer with probability greater than $\frac{1}{n+1}$ then this relation can also be computed by a Las Vegas transducer with probability $\frac{1}{n}$.

Here we show that there exists an infinite hierarchy of relations that can be computed with different probabilities.

Theorem 2. *For every n there exists a left-total relation R_n such that it can be computed by a Las Vegas transducer with probability $\frac{1}{n}$, but there exists no Las Vegas transducer that computes R_n with a probability exceeding $\frac{1}{n}$.*

Proof. Indeed, consider the following relation:

$$R_n = \{(x, y) | x, y \in \{1\}^*, y = 1^{|x| \bmod n}\}$$

First let us prove that it can be solved with probability $p = \frac{1}{n}$:

We will build a transducer for this relation which will consist of the initial state, from which n transitions emerge, each with the probability $\frac{1}{n}$, and n *cycles* of n states (with each state in the cycle outputting nothing and always leading only to the next, except the n-th state, which leads to the first state of the cycle) of which only one state is the final. Let transitions coming from the initial state be numbered from 1 to n, and i be this number, then each transition outputs i 1s (except for $i = n$, which outputs nothing), and ends up in the first state of it's *cycle*. In each cycle, only the i-th state is the final. (In essence, we compute this relation with the probability $\frac{1}{n}$, by simply *guessing* which of the n possible values ends up being.)

Now let us prove that it can not be solved with probability $p' > \frac{1}{n}$:

First, we will assume that there exists a transducer that can achieve that. Then with $|x|$ being $(n-1)$ it has to output $(n-1)$ 1s with the probability $p' > \frac{1}{n}$. However if we now give it a longer input word once $|x|$ becomes $(2n-2)$ the output has to be only $(n-2)$ 1s with probability at least $p' > \frac{1}{n}$. This process can be repeated n times, until $|x|$ becomes $(n^2 - n)$, and the output has to be empty, with probability at least $p' > \frac{1}{n}$. Since none of these events can occur simultaneously (remember that the transducer cannot erase symbols from the output tape), the union of their probabilities is $n \cdot p' > \frac{n}{n} = 1$, which is impossible, hence a contradiction, and it's proved that no such transducer can exist. □

Here we show how nondeterministic transducers compare to Las Vegas transducers.

It is easy to construct a nondeterministic transducer from a Las Vegas transducer – one only has to remove probabilities from the transitions. However, as

the next theorem shows, in some cases it is not possible to do this in the opposite direction therefore nondeterministic transducers are more powerful than Las Vegas transducers.

Theorem 3. *There exists a left-total relation R such that R can be computed by a nondeterministic transducer, but R cannot be computed by any Las Vegas transducer with any fixed probability exceeding* 0.

Proof. Consider the following relation:

$$R = \left\{ (x, y) \,\middle|\, x, y \in \{0,1\}^*, x = 0^{a_1}10^{a_2}1 \ldots 10^{a_k}, y = 0^{f(a_1)}10^{f(a_2)}1 \ldots 10^{f(a_k)} \right\}$$

where

$$f(n) = \begin{cases} n, & \text{if } n \equiv 0 \,(\mathrm{mod} 2) \\ 2n, & \text{if } n \equiv 1 \,(\mathrm{mod} 2) \end{cases}$$

Blocks of zeros (length of the block can be 0) have to be transformed to the same block if the length of the block is even and to a block twice the length if the size of the block is odd.

A nondeterministic transducer can "guess" the parity of the size of each block and if the guess was correct then go to an accepting state, otherwise stay at a non-accepting state.

However no Las Vegas transducer can compute this relation. We can probabilistically construct an input word $x_n = 0^{X_1}10^{X_2}1 \ldots 10^{X_n}$ where

$$X_i = \begin{cases} i, & \text{with probability } \frac{1}{2} \\ i+1, & \text{with probability } \frac{1}{2} \end{cases}$$

It can be seen that if i is large enough (larger than the number of states of the transducer) any Las Vegas transducer has to start writing an answer before it has finished reading block 0^{X_i}. So it has to guess the parity of X_i and after m such guesses the probability of giving the correct answer cannot be greater than $\frac{1}{2^m}$. Therefore for every fixed Las Vegas transducer T and fixed probability p, we can pick n large enough that T gives the correct answer on x_n with probability less than p. □

4 Deterministic transducers with help symbols

In this section we will show how Las Vegas transducers compare to another model of computation – deterministic transducers with help symbols.

Here we show how to construct a Las Vegas transducer from a deterministic transducer with help symbols.

Theorem 4. *If a left-total relation R can be computed by a deterministic transducer with the help of n symbols then R can be computed by a Las Vegas transducer with probability $\frac{1}{n}$.*

Proof. Suppose that a deterministic transducer T computes the relation R with the help of n symbols. We can construct a Las Vegas transducer T' that computes R with probability $\frac{1}{n}$ in the following way.

We make n copies of T such that the i-th copy contains only the transitions with i as the help symbol. Then we add an initial state for the transducer T' from which we add n ε-transitions each with probability $\frac{1}{n}$ to the initial states of each of the copies of T.

It is easy to see that with a probability $\frac{1}{n}$ the transducer guesses the correct help symbol and therefore computes the correct relation. □

We believe that this is true also in other direction, i.e. that every Las Vegas transducer can be transformed into an equivalent deterministic transducer with help symbols.

Conjecture 2. If a left-total relation R can be computed by a Las Vegas transducer with probability $\frac{1}{n}$ then R can be computed by a deterministic transducer with a help of n symbols.

For deterministic transducers with help symbols there also is a infinite hierarchy of relations.

Theorem 5. *For every n there exists a left-total relation R_n such that R_n can be computed by a deterministic transducer with the help of n symbols, but cannot be computed with the help of $n-1$ symbols or less.*

Proof. Denote by R_n the relation of Theorem 2. This relation can be computed by a Las Vegas transducer with probability of $\frac{1}{n}$, but cannot be computed with a probability greater than $\frac{1}{n}$.

We now claim that this relation indeed satisfies the required properties:

- R_n can be computed with a deterministic transducer with the help of n symbols. The help symbol for input word x will be $|x| \bmod n$. Having received the help symbol the transducer will print $1^{|x| \bmod n}$ and then loop accepting any input of the required length.
- assume from the contrary that the deterministic transducer D computes R_n with the help of $n-1$ symbols or less. Then by the Theorem 4 there is a Las Vegas transducer that computes R_n with probability $\frac{1}{n-1}$. This contradicts our choice of R_n, therefore our assumption is false and R_n cannot be computed with the help of $n-1$ symbols or less.

□

5 Frequency transducers

First of all, it should be noted that a frequency transducer does not specify uniquely the relation computed by it. For instance, a transducer with 3 input tapes and 3 output tapes $(2,3)$-computing the relation

$$R(x,y) = \begin{cases} \text{true, if } x = y \text{ and } x \neq 258714, \\ \text{true, if } y = 0 \text{ and } x = 258714, \\ \text{false, if otherwise.} \end{cases}$$

can output $y = x$ for all possible inputs and, nonetheless, the result is always correct for at least 2 out of 3 inputs since the inputs always distinct. Please notice that the program of the frequency transducer does not contain the "magical" number 258714. Hence the number of states for an equivalent deterministic transducer can be enormously larger.

Theorem 6. *There exists a strong finite-state frequency transducer* $(1, 2)$*-computing a continuum of left-total relations.*

Proof. Consider the following frequency transducer with 2 input tapes and 2 output tapes. If the inputs x_1 and x_2 are such that x_1 is an initial fragment of x_2, then the output y_1 equals x_1, and $y_2 = 0$. If the inputs x_1 and x_2 are such that x_2 is an initial fragment of x_1, then the output y_2 equals x_2, and $y_1 = 0$. In all the other cases $y_1 = y_2 = 0$.

Let R be a relation defined by taking an arbitrary infinite sequence ω of zeros and ones, and defining

$$R(x,y) = \begin{cases} \text{true, if } y = x \text{ and } x \text{ is an initial fragment of } \omega, \\ \text{true, if } y = 0 \text{ and } x \text{ is not an initial fragment of } \omega, \\ \text{false, if} \qquad\qquad \text{otherwise.} \end{cases}$$

There is a continuum of such sequences and a continuum of the corresponding relations. Each of them is $(1, 2)$-computed by the frequency transducer. □

Theorem 7. *If* $2m > n$ *and there exists a strong finite-state frequency transducer* (m, n)*-computing a left-total relation* R *then* R *can also be computed by a deterministic finite-state transducer.*

Proof. Let R be a left-total relation and A be a strong finite-state frequency transducer (m, n)-computing it. Let Q be (another) left-total relation (m, n)-computed by the same transducer A. Let $x_1, x_2, \cdots, x_k$ be distinct input words such that the pairs $(x_1, y_1'), (x_2, y_2'), \cdots, (x_k, y_k')$ are in R, the pairs (x_1, y_1''), $(x_2, y_2''), \cdots, (x_k, y_k'')$ are in Q, and $y_1' \neq y_1'', y_2' \neq y_2'', \cdots, y_k' \neq y_k''$. What happens if $k \geq n$ and the transducer gets an input-tuple of distinct values containing only values from $\{x_1, x_2, \cdots, x_k\}$? The transducer has to output at least m correct values for R and at least m correct values for Q which is impossible because $2m > n$. A careful count shows that $k \leq 2n - 2m$.

Hence for arbitrary given relation R (m, n)-computable by A there is a constant k such that any relation Q (m, n)-computable by A differs from R on no more that k pairs (x, y). Let Q_0 be one of the relations where indeed k such pairs exist, and let $(x_1, y_1'), (x_2, y_2'), \cdots, (x_k, y_k')$ and (x_1, y_1''), $(x_2, y_2''), \cdots, (x_k, y_k'')$ be these pairs. Of course, there is no algorithm to construct k and Q_0 effectively, but they cannot fail to exist. For arbitrary x the value of y such that $(x, y) \in R$ can be calculated effectively using n-tuples of input words involving the input words $x_1, x_2, \cdots, x_k$ and remembering that no relation computed by A can differ from R and from Q_0 in more than k values. Since A is a finite automaton, the standard cut-and-paste arguments show that this computation needs only input words which differ from x only on the last d digits where d is a suitable constant. □

Theorem 8. *There exists a left-total relation R $(2,3)$-computed by a weak finite-state Las Vegas frequency transducer and not computed by any deterministic finite-state transducer.*

Proof. We consider the relation

$$R(x,y) = \begin{cases} \text{true, if } y = 1^{|x|} & \text{and } \mid x \mid \equiv 0(mod3), \\ \text{true, if } y = 1^{|x|} & \text{and } \mid x \mid \equiv 1(mod3) \\ \text{true, if } y = 1^{|x|-1} & \text{and } \mid x \mid \equiv 2(mod3) \\ \text{false, if} & \text{otherwise.} \end{cases}$$

The weak finite-state Las Vegas frequency $(2,3)$-transducer starts with reading 1 symbol from the first input tape, 2 symbols from the second input tape and 3 symbols from the third input tape and reads exactly 1 symbol from each input tape at any moment. Hence at any moment the transducer has no more than one input word x_i with $\mid x_i \mid \equiv 2(mod3)$. To have a correct result for the other two input words, it suffices to keep the length of the output being equal the length of the corresponding input. In case if the length of the input is $\mid x_i \mid \equiv 2(mod3)$, the state becomes non accepting.

The relation cannot be computed by a deterministic transducer because the length of the output y decreases when the length of of the input increases from $3k+1$ to $3k+2$. □

This proof can easily be extended to prove the following Theorem 9.

Theorem 9. *If $m < n$ then there exists a left-total relation R (m,n)-computed by a weak finite-state Las Vegas frequency transducer and not computed by any deterministic finite-state transducer.*

Theorem 10. *There exists a left-total relation R $(2,4)$-computed by a weak finite-state Las Vegas frequency transducer and not computed by any finite-state $(1,2)$-frequency transducer.*

Proof. We consider the relation

$$R(x,y) = \begin{cases} \text{true, if } y = 1^{|x|} & \text{and } \mid x \mid \equiv 0(mod9), \\ \text{true, if } y = 1^{|x|} & \text{and } \mid x \mid \equiv 3(mod9), \\ \text{true, if } y = 1^{|x|} & \text{and } \mid x \mid \equiv 4(mod9), \\ \text{true, if } y = 1^{|x|} & \text{and } \mid x \mid \equiv 6(mod9), \\ \text{true, if } y = 1^{|x|} & \text{and } \mid x \mid \equiv 8(mod9), \\ \text{true, if } y = 0 & \text{and } \mid x \mid \equiv 1(mod9), \\ \text{true, if } y = 0 & \text{and } \mid x \mid \equiv 2(mod9), \\ \text{true, if } y = 0 & \text{and } \mid x \mid \equiv 5(mod9), \\ \text{true, if } y = 0 & \text{and } \mid x \mid \equiv 7(mod9), \\ \text{false, if} & \text{otherwise.} \end{cases}$$

(1) The weak finite-state Las Vegas frequency $(2,4)$-transducer starts with reading 1 symbol from the first input tape, 2 symbols from the second input tape

and 3 symbols from the third input tape, 4 symbols from fourth input tape and reads exactly 1 symbol from each input tape at any moment. The transducer always outputs $y_i = 1^{|x_i|}$ on the i-th output tape. Since the transducer can count the length of the input modulo 9, the false outputs (in cases $\mid x_i \mid$ congruent to 1,2,5 or 7 (mod 9)) are not accepted and the transducer is Las Vegas.

At every moment the lengths of the input words are $k, k+1, k+2, k+3$ for some natural k. At least two of them are congruent to 0,3,4,6 or 8 (mod 9).

(2) Assume that the relation is $(1,2)$-computed by a transducer performing a *weak* $(1,2)$-computation with parameters (b_1, b_2). Whatever the difference $d = b_2 - b_1$, there exists a value of s such that both $s + b_1$ and $s + b_2$ are congruent to 1,2,5 or 7 (mod 9). Hence the transducer produces two wrong results on the pair $1^{s+b_1}, 1^{s+b_2}$ in contradiction with the $(1,2)$-computability. □

References

1. Holger Austinat, Volker Diekert, Ulrich Hertrampf, Holger Petersen. Regular frequency computations. *Theoretical Computer Science,* vol. 330 No. 1, pp. 15–20, 2005.
2. Richard Beigel, William I. Gasarch, Efim B. Kinber. Frequency computation and bounded queries. *Theoretical Computer Science,* vol. 163 No. 1/2, pp. 177–192, 1996.
3. John Case, Susanne Kaufmann, Efim B. Kinber, Martin Kummer. Learning recursive functions from approximations. *Journal of Computer and System Sciences,* vol. 55, No. 1, pp. 183–196, 1997.
4. A.N.Degtev. On (m,n)-computable sets. *Algebraic Systems,* Edited by D.I. Moldavanskij, Ivanovo Gos. Universitet, pp. 88–99, 1981.
5. Eitan Gurari. An Introduction to the Theory of Computation. *Computer Science Press, an imprint of E. H. Freeman,* Chapter 2.2, 1989.
6. Rūsiņš Freivalds. Amount of nonconstructivity in finite automata. *Theoretical Computer Science,* vol. 411, No. 38-39, pp.3436–3443, 2010.
7. Valentina Harizanova, Martin Kummer, Jim Owings. Frequency computations and the cardinality theorem. *The Journal of Symbolic Logic,* vol. 57, No. 2, pp. 682–687, 1992.
8. Maren Hinrichs and Gerd Wechsung. Time bounded frequency computations. *Information and Computation,* vol. 139, pp. 234-257, 1997.
9. Efim B. Kinber. Frequency calculations of general recursive predicates and frequency enumeration of sets. *Soviet Mathematics Doklady,* vol. 13, pp. 873–876, 1972.
10. Efim B. Kinber. Frequency computations in finite automata, *Kibernetika,* No. 2, pp. 7–15, 1976(Russian; English translation in Cybernetics 12 (1976) 179-187).
11. Martin Kummer. A proof of Beigel's Cardinality Conjecture. *The Journal of Symbolic Logic,* vol. 57, No. 2, pp. 677–681, 1992.
12. Robert McNaughton. The Theory of Automata, a Survey. *Advances in Computers,* vol. 2, pp. 379–421, 1961.
13. Gene F. Rose. An extended notion of computability. *Abstracts of International Congress for Logic, Methodology and Philosophy of Science,* p.14, 1960.
14. Boris A. Trakhtenbrot. On the frequency computation of functions. *Algebra i Logika,* vol. 2, pp.25–32, 1964 (Russian)