Universal Simulation of Automata Networks

Alonso Castillo-Ramirez and Maximilien Gadouleau

School of Engineering and Computing Sciences,
Durham University, South Road,
Durham, DH1 3LE

April 29, 2015

Abstract

Let $A$ be a finite set and $n \geq 2$. This paper introduces the concept of universal simulation in the context of semigroups of transformations of $A^n$, also known as finite state-homogeneous automata networks. For $m \geq n$, a transformation of $A^n$ is defined as $n$-universal of size $m$ if it may simulate every transformation of $A^n$ by updating one coordinate (or register) at a time. Using tools from memoryless computation, it is established that there is no $n$-universal transformation of size $n$, but there is such a transformation of size $n + 2$. An $n$-universal transformation is defined as complete if it may sequentially simulate every finite sequence of transformations of $A^n$; in this case, minimal examples and bounds for the size and time of simulation are determined. It is also shown that there is no $n$-universal transformation that updates all the registers in parallel, but that there exists a complete one that updates all but one register in parallel. This illustrates the strengths and weaknesses of parallel models of computation, such as cellular automata.

1 Introduction

Memoryless computation (MC) is a modern paradigm for computing any transformation of $A^n$, with $A$ a finite set and $n \geq 2$, by updating one coordinate at a time while using no memory. Its basic idea was developed in [2, 3, 4, 5, 6, 7, 8], and expanded in [9, 10, 12]. The seminal example of MC is the famous XOR swap algorithm, which is analogous to the butterfly network, the canonical example of network coding (see [1]). In the following paragraphs, we shall introduce notation and review the main definitions of MC.

Let $q$ be the cardinality of $A$. Without loss, we usually regard $A$ as the ring $Z_q = Z/qZ$ or, when $q$ is a prime power, the field $GF(q)$. Since the cases when $q = 1$ or $n = 1$ are trivial, we shall assume $q \geq 2$ and $n \geq 2$ henceforth. We refer the coordinates of $A^n$ as registers and the elements of $A^n$ as states. Denote by $e^k \in A^n$ the state with 1 at its $k$-th register and zero everywhere else, and by $e^0 \in A^n$ the state with zeros in all its registers. For any $a \in A^n$, we denote by $a_i$ the image of $a$ under the $i$-th coordinate projection.

We are interested in studying transformations of $A^n$, i.e., functions of the form $A^n \rightarrow A^n$. Denote by $\text{Tran}(A^n)$ the set of all transformations of $A^n$, and by $\text{Sing}(A^n)$ and $\text{Sym}(A^n)$ the set of all singular and nonsingular transformations of $A^n$, respectively. The sets $\text{Tran}(A^n)$ and $\text{Sing}(A^n)$, equipped with the composition of transformations $\circ$, form semigroups called the full transformation semigroup on $A^n$ and the singular semigroup on $A^n$, respectively. The set $\text{Sym}(A^n)$, equipped with $\circ$, forms a group called the symmetric group on $A^n$.

In general, if $Y$ is a subset of a semigroup $S$, let $\langle Y \rangle$ be the smallest subsemigroup of $S$ containing $Y$. Say that $Y$ is a generating set of $S$ when $S = \langle Y \rangle$. In particular, if $S$ is a subsemigroup of $\text{Tran}(A^n)$, and $Y$ is a generating set of $S$, the triple $(A^n, S, Y)$ is referred as a finite state-homogeneous automata network (see [11] p. 200).

Throughout this paper, we adopt the convention of applying functions on the right; then $(x)f$ denotes the image of $x \in A^n$ under $f \in \text{Tran}(A^n)$, and $f \circ g$ (or simply $fg$) denotes the composition of functions $(x)(f \circ g) = ((x)f)g$. The size of the image of a transformation is referred as its rank.
We view each transformation of $A^n$ as a tuple of functions $f = (f_1, \ldots, f_n)$, where $f_i : A^n \rightarrow A$ is referred to as the $i$-th coordinate function of $f$. In particular, an $i$-th coordinate function is trivial if it is equal to the $i$-th projection: $(x)f_i = x_i$, for all $x \in A^n$. When considering a sequence of transformations, we shall use superscripts in brackets, e.g. $f^{(1)}, \ldots, f^{(\ell)} \in \text{Tran}(A^n)$.

The following is the key definition of memoryless computation.

**Definition 1** (Instruction). An instruction of $A^n$ is a transformation $f : A^n \rightarrow A^n$ with at most one nontrivial coordinate function. A permutation instruction is an instruction which maps $A^n$ bijectively onto $A^n$.

The previous definition implies that the identity transformation of $A^n$ is an instruction. We denote the set of instructions of $A^n$ as $\mathcal{I}(A^n)$, and the set of permutation instructions as $\mathcal{I}(A^n)$. We shall simply write $\mathcal{I}$ and $\mathcal{I}$ when there is no ambiguity. Note that any nontrivial instruction $f \in \mathcal{I}$ is uniquely determined by its nontrivial coordinate function $f_i$; hence, in this case, we say that $f$ updates the $i$-th register, and we shall often denote $f$ by its update form:

$$f : x_i \leftarrow (x)f_i.$$ 

For instance, if $A = \text{GF}(2)$ and $n = 2$, then $\mathcal{I}$ is given by

$$\{x_1 \leftarrow x_1, \ x_1 \leftarrow x_1 + 1, \ x_1 \leftarrow x_1 + x_2, \ x_1 \leftarrow x_1 + x_2 + 1, \ x_2 \leftarrow x_2, \ x_2 \leftarrow x_2 + 1, \ x_2 \leftarrow x_1 + x_2, \ x_2 \leftarrow x_1 + x_2 + 1\},$$

where the identity may be represented by either $x_1 \leftarrow x_1$ or $x_2 \leftarrow x_2$.

One of the most important features of the instruction sets $\mathcal{I}$ and $\mathcal{I}$ is that they are generating sets of $\text{Tran}(A^n)$ and $\text{Sym}(A^n)$, respectively (see [2][12]).

**Definition 2** (Program). For any $g \in \text{Tran}(A^n)$, a program of length $\ell$ computing $g$ is a sequence of instructions $f^{(1)}, \ldots, f^{(\ell)} \in \mathcal{I}$ such that

$$g = f^{(1)} \circ \ldots \circ f^{(\ell)}.$$

Unless specified otherwise, we assume that every instruction in a program is different from the identity. Moreover, since the set of instructions updating a given register is closed under composition, we may always assume that $f^{(k+1)}$ updates a different register than $f^{(k)}$ for all $k$. In this paper, we shall work with particular subsets of instructions $Y \subseteq \mathcal{I}$. Hence, for any transformation $g \in \langle Y \rangle$, we define the memoryless complexity of $g$ with respect to $Y$ as the minimum length of a program computing $g$ with instructions from $Y$. The memoryless complexity of $g$ with respect to $\mathcal{I}$ is simply called the memoryless complexity of $g$.

**Example 1.** In order to illustrate our notations, let us write the program computing the swap of two variables, i.e. $g : \mathbb{Z}_q^2 \rightarrow \mathbb{Z}_q^2$ where $(x_1, x_2)g = (x_2, x_1)$. It is given as follows:

$$g = f^{(1)} \circ f^{(2)} \circ f^{(3)},$$

where

$$f^{(1)} : x_1 \leftarrow x_1 + x_2,$$

$$f^{(2)} : x_2 \leftarrow x_1 - x_2,$$

$$f^{(3)} : x_1 \leftarrow x_1 - x_2,$$

or, equivalently

$$(x_1, x_2)f^{(1)} = (x_1 + x_2, x_2),$$

$$(x_1, x_2)f^{(2)} = (x_1, x_1 - x_2),$$

$$(x_1, x_2)f^{(3)} = (x_1 - x_2, x_2).$$
This paper is organised as follows. In Section 2, we introduce our notion of simulation, which is a way of computing a transformation of \( A^n \) using \( m \geq n \) instructions that may depend on \( m - n \) extra registers. We say that a transformation of \( A^n \) is \( n \)-universal if the instructions induced by its coordinate functions may simulate any transformation of \( A^n \). We show that there is no \( n \)-universal transformation that uses no extra registers, but that there is one that uses only two extra registers. Then, we construct an \( n \)-universal transformation with maximal time of simulation \( q^n + O(n) \), and conjecture that \( q^n \) is the lower bound for the maximal time of simulation of any \( n \)-universal transformation.

In Section 3, we introduce the notion of sequential simulation. We say that an \( n \)-universal transformation is complete if it may sequentially simulate any sequence of transformations of \( A^n \). We establish that any such transformation requires at least \( n \) extra registers, and we construct one with \( n + 2 \) extra registers when \( q \geq 3 \), and \( n + 3 \) extra registers when \( q = 2 \). Then, we establish lower bounds for the maximal and minimal time of simulation of complete \( n \)-universal transformations, and construct explicit examples that asymptotically tend to these bounds.

Finally, in Section 4, we show that there is no universal transformation that updates all the registers in parallel; however, we construct a complete \( n \)-universal transformation that updates all but one register in parallel. The first result shows that some asynchronism is required in order to obtain universality; conversely, the second result shows that the least amount of asynchronism is enough to obtain universality.

Our work in this paper differentiates in several aspects from results on universal cellular automata (see e.g. [13, 15]). One of these main differences is that simulation on cellular automata usually assumes an initial configuration of the input states. On the other hand, our work requires no initial configuration: an \( n \)-universal transformation is able to simulate every transformation using any configuration of the states of \( A^m \) as inputs. This is illustrated by the results in Section 4, especially in the complete \( n \)-universal transformation that updates all but one register in parallel. Indeed, this transformation only uses asynchronism to reset a counter, i.e. to place the state in a special initial configuration; once this is done, the parallel updates are then complete \( n \)-universal.

## 2 Simulation of transformations

Denote \([n] := \{1, \ldots, n\}\). For \( m \geq n \), let \( \text{pr}_{[n]} : A^m \to A^n \) be the \([n]\)-projection of \( A^m \) to \( A^n \), i.e., \((x_1, \ldots, x_m) \text{pr}_{[n]} = (x_1, \ldots, x_n)\). For any \( f : A^m \to A^n \), define

\[
S_f := \{F^{(1)}, \ldots, F^{(m)}\} \leq \text{Tran}(A^n),
\]

where \( F^{(i)} : A^m \to A^n \) is the instruction induced by the coordinate function \( f_i \):

\[
F^{(i)} : x_i \leftarrow (x)f_i.
\]

### Definition 3 (Simulation).

Let \( m \geq n \geq 2 \). We say that \( f : A^m \to A^m \) simulates \( g : A^n \to A^n \) if there exists \( h \in S_f \) such that \( \text{pr}_{[n]} \circ g = h \circ \text{pr}_{[n]} \). The time of simulation, denoted by \( t_f(g) \), is the memoryless complexity of \( h \) with respect to \( \{F^{(1)}, \ldots, F^{(m)}\} \).

Compare our previous definition of simulation with the definition of *simulation by projection* for finite state-homogeneous automata networks that appears in [11, p. 208].

### Definition 4 (\(n\)-Universal).

Let \( m \geq n \geq 2 \). A transformation \( f : A^m \to A^m \) is called \( n \)-universal of size \( m \) if it may simulate any transformation in \( \text{Tran}(A^n) \). The time of \( f \) is \( t_f := \max \{t_f(g) : g \in \text{Tran}(A^n)\} \).

### Example 2.

For any \( n \geq 2 \), there is an elementary example of an \( n \)-universal transformation \( f : A^m \to A^m \), with size \( m = 2n + 2q^{n^2} \). In order to describe it, we first let \( Q := q^n \) and enumerate by \( p^{(s)} \), \( 1 \leq s \leq Q \), all the transformations in \( \text{Tran}(A^n) \). For \( 1 \leq i \leq n \), define the coordinate functions \( f_i \) of \( f \) as follows:

\[
(f)_{f_i} = \begin{cases} 
(x) \text{pr}_{[2n]\{[n]\}} \circ p^{(s)}_i & \text{if } x_{2n+r} \neq x_{2n+Q^{n+r}} \text{ and } x_{2n+r} = x_{2n+Q^{n+r}} \quad (\forall 1 \leq r \leq Q^n, r \neq s) \\
(x_i & \text{otherwise.}
\end{cases}
\]
The rest of the coordinate functions of $f$ are:

$$(x)f_r = x_{r-n}, \quad (n+1 \leq r \leq 2n)$$
$$(x)f_j = x^{Q^n+j}, \quad (2n+1 \leq j \leq 2n+Q^n)$$
$$(x)f_k = x_k + 1, \quad (2n+Q^n + 1 \leq k \leq 2n + 2Q^n).$$

The main idea in this transformation is to use $Q^n$ switches, each one corresponding to a possible value of $s$ (and hence describing the transformation $p^{(s)}$ that we are simulating) and consisting of two registers: $2n + s$ and $2n + Q^n + s$.

To show that $f$ is indeed an $n$-universal transformation, let $F^{(i)}$ be the instruction induced by the coordinate function $f_i$. Suppose that we want to simulate $p^{(s)}$, $1 \leq s \leq Q^n$. Then, this may be achieved as follows.

**Step 1.** Make a copy the first $n$ registers: $F^{(n+1)} \circ \cdots \circ F^{(2n)}$.

**Step 2.** Turn all switches off: $F^{(2n+1)} \circ \cdots \circ F^{(2n+Q^n)}$.

**Step 3.** Turn the right switch on: $F^{(2n+Q^n+s)}$.

**Step 4.** Compute $p^{(s)}$: $F^{(1)} \circ \cdots \circ F^{(n)}$.

Or more concisely, the transformation

$$h := (F^{(n+1)} \circ \cdots \circ F^{(2n)}) \circ (F^{(2n+1)} \circ \cdots \circ F^{(2n+Q^n)}) \circ F^{(2n+Q^n+s)} \circ (F^{(1)} \circ \cdots \circ F^{(n)})$$

satisfies $pr_{[n]} \circ p^{(s)} = h \circ pr_{[n]}$. As such, the time of $f$ is $t_f = n + Q^n + 1 + n = q^nq^n + O(n)$.

In the following sections, we study $n$-universal transformations with minimal size and time.

### 2.1 Universal transformations of minimal size

In this section, we denote the transposition of $u, v \in A^n$ as $(u, v)$, where, for any $x \in A^n$,

$$(x)(u, v) = \begin{cases} 
u & \text{if } x = u \\ u & \text{if } x = v \\ x & \text{otherwise}, \end{cases}$$

and the assignment of $u$ to $v$ as $(u \rightarrow v)$, where

$$(x)(u \rightarrow v) = \begin{cases} 
u & \text{if } x = u \\ x & \text{otherwise}. \end{cases}$$

For any $f \in \text{Tran}(A^n)$ and $g \in \text{Sym}(A^n)$, the conjugation of $f$ by $g$ is $f^g := g^{-1}fg \in \text{Tran}(A^n)$.

It was determined in [10] that, unless $|A| = n = 2$, there exists a set $Y \subset \mathcal{I}$ of size $n$ that generates the whole symmetric group $\text{Sym}(A^n)$: hence, the set $Y \cup \{(e^0 \rightarrow e^1)\}$ of $n+1$ instructions suffice to generate the full transformation semigroup $\text{Tran}(A^n)$. In the following theorem, we prove there is no set of $n$ instructions that generate $\text{Tran}(A^n)$, which implies that there is no $n$-universal transformation of size $n$.

**Theorem 1.** For any $n \geq 2$, there is no transformation $f \in \text{Tran}(A^n)$ such that $\text{Sing}(A^n) \leq S_f$.

**Proof.** Suppose that $Y := \{F^{(1)}, \ldots, F^{(n)}\}$ is a set of instructions that generate a semigroup containing all singular transformations, where $F^{(i)}$ updates the $i$-th register. Since the composition of permutations is a permutation, at least one of these generating instructions must be singular.

First, assume that at least two instructions of $Y$, say $F^{(1)}$ and $F^{(2)}$, are singular. We claim that no assignment $g = (a \rightarrow b)$, with $a_i \neq b_i$, $i = 1, 2$, can be computed using only instructions in $Y$. Indeed, suppose that $F^{(1)}$ is the first singular instruction in a program computing $g$, so $g = \pi \circ F^{(1)} \circ h$, for some $h \in \text{Tran}(A^n)$ and $\pi \in \{F^{(3)}, \ldots, F^{(n)}\}$. As $\pi \circ F^{(1)}$ is singular, there exist $u, v \in A^n$, $u \neq v$, such
that \((u)\pi \circ F^{(1)} = (v)\pi \circ F^{(1)}\), which implies that \((u)g = (v)g\). However, as \(\pi \circ F^{(1)}\) does not update the second register, we have \(\{u, v\} \neq \{a, b\}\), which contradicts the definition of the assignment \(g\).

By the previous paragraph, there may be only one singular instruction in \(Y\), say \(F^{(1)}\). Let \(u, v \in A^n\), \(u_1 \neq v_1\), be such that \((u)F^{(1)} = (v)F^{(1)}\). For any \(g \in \text{Sing}(A^n)\), we may write \(g = \pi \circ F^{(1)} \circ h\), where \(h \in \text{Tran}(A^n)\) and \(\pi \in \langle F^{(2)}, \ldots, F^{(n)} \rangle \subseteq \text{Sym}(A^n)\). Letting \(x = (u)\pi^{-1}\) and \(y = (v)\pi^{-1}\), we see that \(x_1 \neq y_1\) and \((x)g = (y)g\). However, this means that assignments such as \(g = (a \rightarrow b)\), with \(a \neq b\), cannot be computed.

**Corollary 1.** For any \(n \geq 2\), there is no \(n\)-universal transformation of size \(n\).

It is an open question whether there is an \(n\)-universal transformation of size \(n + 1\). However, we shall find an \(n\)-universal transformation of size \(n + 2\) and time at most \(6[\log_2(q)](q - 1)q^{n - 1} + O(q^n)\). Before this, we need the following result of memoryless computation.

**Theorem 2.** Let \(|A| = q\) and \(n \geq 2\). Then \(\text{Tran}(A^n)\) is generated by a set of instructions \(Y\), containing at most \(q\) instructions per register, such that any transformation of \(A^n\) has memoryless complexity with respect to \(Y\) of at most \(3[\log_2(q)](q - 1)q^{n - 1} + O(q^n)\).

**Proof.** We consider the following instructions:

\[
T^{(1)}: \quad x_1 \leftarrow x_1 + \delta(x, e^0) - \delta(x, e^1),
\]

\[
A^{(2)}: \quad x_2 \leftarrow x_2 + \delta(x, e^0),
\]

\[
I^{(1)}: \quad x_1 \leftarrow x_1 + 1 - \delta(x, e^0) + \delta(x, (q - 1)e^1),
\]

\[
I^{(i)}: \quad x_i \leftarrow x_i + 1 - \sum_{\lambda \in A} \delta(x, \lambda e^i),
\]

(for \(2 \leq i \leq n\)),

where \(\delta(x, y)\) denotes the Kronecker delta function, and \(\lambda e^i\) is the state with \(\lambda \in A\) in its \(i\)-th register and zero elsewhere. In order to simplify notation, we shall identify \(x \in A^n\) with its lexicographic index \(\sum_{i=1}^n x_i q^{i-1} \in \{0, 1, \ldots, q^n - 1\}\). With this, we may write \(A^{(2)} = (0 \rightarrow q)\) and \(T^{(1)} = (0, 1)\).

Observe that the instructions \(I^{(i)}\) are permutations with the following cyclic structure: \(I^{(1)}\) consists of one cycle of length \(q - 1\) and \(q^{n-1} - 1\) cycles of length \(q\), while, for \(2 \leq i \leq n\), the instruction \(I^{(i)}\) consist of just \(q^{n-1} - 1\) cycles of length \(q\).

Let \(\rho := \lceil \log_2(q) \rceil\) and define

\[Y := \left\{ T^{(1)}, A^{(2)}, (I^{(i)})^{2^j} : 1 \leq i \leq n, \ 0 \leq j \leq \rho - 1 \right\}.\]

We shall follow several steps in order to prove that \(Y\) is the required generating set.

(i) Any transposition \(T^{(k)} := (0, k)\), with \(k \in A^n\), has memoryless complexity with respect to \(Y\) of at most \(pw(k) + O(1)\), where \(w(k)\) is the number of non-zero coordinates of \(k\).

**Proof.** First, we determine the memoryless complexity of \((I^{(i)})^\lambda\), for \(1 \leq i \leq n\) and \(1 \leq \lambda \leq q - 1\) with respect to \(Y\). Using the binary expansion \(\lambda = \sum_{j=1}^{\rho} \lambda_j 2^{j-1}\), \(\lambda_j \in \{0, 1\}\), it is clear that

\[(I^{(i)})^\lambda = (I^{(i)})^{\lambda_1} \circ ((I^{(i)})^2)^{\lambda_2} \circ \cdots \circ ((I^{(i)})^{2^{\rho-1}})^{\lambda_\rho}.\]

Thus, we need at most \(\rho\) instructions from \(Y\) to compute \((I^{(i)})^\lambda\).

Fix \(k \in A^n\), and suppose that \(1 \leq j_1, \ldots, j_w \leq n\), with \(w = w(k)\), are the non-zero coordinates of \(k\). If \(k\) is not a multiple of \(q\) (i.e. \(j_1 = 1\)), we have

\[T^{(k)} := (0, k) = \left(T^{(1)}\right)^{(I^{(1)})^{k_1 - 1}(I^{(j_2)})^{k_2} \ldots (I^{(j_w)})^{k_w}},\]

while if \(k\) is a multiple of \(q\), we have

\[T^{(k)} = \left(T^{(1)}\right)^{(I^{(j_1)})^{k_{j_1}} \ldots (I^{(j_w)})^{k_{j_w}} (I^{(1)})^{q-1}}.\]

The result follows because \((I^{(i)})^{-1} = (I^{(i)})^{q-\lambda}\), for any \(1 \leq \lambda \leq q - 1\) and \(2 \leq i \leq n\). \(\square\)
(ii) Any permutation in $\text{Tran}(A^n)$ has memoryless complexity with respect to $Y$ of at most $2(q-1)nstq^n+O(q^n)$.

**Proof.** Note that any transposition $(a,b)$ may be expressed as

$$w((a,b)) = T^{(b)} T^{(a)} T^{(b)}.$$  

Since any permutation with $r$ non-fixed points may be expressed as at most $r-1$ transpositions, cyclic permutations of length $q^n$ have the maximum memoryless complexity. In particular, if $\pi = (a_1, a_2, \ldots, a_{q^n}) \in \text{Sym}(A^n)$, then

$$\pi = (a_1, a_2) \ldots (a_{q^n-1}, a_{q^n})$$  

$$(T^{(a_2)} T^{(a_3)} \ldots T^{(a_{q^n-1})} T^{(a_{q^n-2})} T^{(a_{q^n-3})} \ldots T^{(a_1)}) = (T^{(a_2)} T^{(a_3)} \ldots T^{(a_{q^n-1})} T^{(a_{q^n-2})} T^{(a_{q^n-3})} \ldots T^{(a_1)})$$

In this decomposition, $T^{(a_1)}$ and $T^{(a_{q^n})}$ appear once, while every other transposition $T^{(a_s)}$, $s \not\in \{1, q^n\}$, appears twice. By step (i), $T^{(a_s)}$ requires at most $\rho w(a_s) + O(1)$ instructions from $Y$. Since

$$\sum_{k \in A^n} w(k) = \sum_{i=1}^{q^n} i(q-1)^i \binom{n}{i} = (q-1)nstq^n$$

it takes at most

$$2\sum_{s=2}^{q^n-1} (\rho w(a_s) + O(1)) \leq 2\rho(q-1)nstq^n + O(q^n)$$

instructions from $Y$ to compute $\pi$. 

(iii) Any transformation in $\text{Tran}(A^n)$ has memoryless complexity with respect to $Y$ of at most $3\rho(q-1)nstq^n+O(q^n)$.

**Proof.** Let $g$ be any transformation of rank $r < q^n$. Consider the partition $\ker(g) := \{P_1, \ldots, P_r\}$ of $A^n$ induced by the following equivalence relation: $a \sim g b$ if and only if $(a)g = (b)g$. (This equivalence relation is called the kernel of $g$.) For $1 \leq i \leq r$, let $P_i = \{p_{i,1}, \ldots, p_{i,r_i}\}$. Depending on two cases, we shall find a transformation $h$ such that $\ker(g) = \ker(h)$, which implies that $g = h \circ \pi$ for some $\pi \in \text{Sym}(A^n)$.

**Case 1:** States 0 and $q$ are in a same set of $\ker(g)$. Without loss, assume $p_{1,1} = 0$ and $p_{1,2} = q$. Then, define

$$h := A^{(2)} T^{(p_{1,3})} A^{(2)} \ldots T^{(p_{1,n_1})} A^{(2)} (q, p_{2,1}) T^{(p_{2,2})} A^{(2)} \ldots T^{(p_{r,n_r})} A^{(2)}.$$  

**Case 2:** States 0 and $q$ are in distinct sets of $\ker(g)$. Without loss, assume $p_{1,1} = 0$ and $p_{r,n_r} = q$. Then, define

$$h := (0, q) T^{(p_{1,2})} A^{(2)} \ldots T^{(p_{1,n_1})} A^{(2)} (q, p_{2,1}) T^{(p_{2,2})} A^{(2)} \ldots T^{(p_{r,n_r-1})} A^{(2)} (p_{j,2}, q),$$

where $j$ is the smallest index for which $p_{j,2}$ exists. (Clearly, such an index $j$ always exists because $g$ does not have full rank.)

Each transposition in $h$ takes at most $\rho w(p_{i,j}) + O(1)$ instructions and each assignment takes $O(1)$ instructions. The result follows by equation (11) and step (ii). 

**Theorem 3.** There exists an $n$-universal transformation of size $n+2$ and time at most

$$6[\log_2(q)](q-1)nstq^n + O(q^n).$$
Proof. Let \( \rho := \lceil \log_2(q) \rceil \). We consider the generating set of instructions \( Y \) given in the proof of Theorem 2. For each instruction in \( Y \), we denote the corresponding nontrivial coordinate function in lowercase, e.g., the nontrivial coordinate function of \( (f^{(2)})^3 \) is \( (x)^3 = x_1 + 2 - 2\delta(x, e^0) + 2\delta(x, (q-1)e^1) \).

Consider the transformation \( f \in \text{Tran}(A^{n+2}) \) with coordinate functions defined as follows:

\[
(x)f_1 = \begin{cases} 
(x)pr_n \circ i_1 & \text{if } x_{n+1} - x_{n+2} = 0 \\
(x)pr_n \circ i_1^2 & \text{if } x_{n+1} - x_{n+2} = 1 \\
\vdots & \vdots \\
(x)pr_n \circ i_{2^{\rho-1}} & \text{if } x_{n+1} - x_{n+2} = \rho - 1 \\
(x)pr_n \circ i_{2^{\rho}} & \text{if } x_{n+1} - x_{n+2} = \rho,
\end{cases}
\]

\[
(x)f_2 = \begin{cases} 
(x)pr_n \circ i_2 & \text{if } x_{n+1} - x_{n+2} = 0 \\
(x)pr_n \circ i_2^2 & \text{if } x_{n+1} - x_{n+2} = 1 \\
\vdots & \vdots \\
(x)pr_n \circ i_{2^{\rho-1}} & \text{if } x_{n+1} - x_{n+2} = \rho - 1 \\
(x)pr_n \circ i_{2^\rho} & \text{if } x_{n+1} - x_{n+2} = \rho,
\end{cases}
\]

\[
(x)f_{j} = \begin{cases} 
(x)pr_n \circ i_j & \text{if } x_{n+1} - x_{n+2} = 0 \\
(x)pr_n \circ i_j^2 & \text{if } x_{n+1} - x_{n+2} = 1 \\
\vdots & \vdots \\
(x)pr_n \circ i_{j^{\rho-1}} & \text{if } x_{n+1} - x_{n+2} = \rho - 1 \\
x_j & \text{if } x_{n+1} - x_{n+2} = \rho,
\end{cases} \quad (3 \leq j \leq n),
\]

\[
(x)f_{n+1} = x_{n+2},
\]

\[
(x)f_{n+2} = x_{n+2} + 1.
\]

The main idea behind the definition of \( f \) is that the additional registers work as a switch to decide which instruction the program shall use.

Let \( F^{(1)} \in \text{Tran}(A^{n+2}) \) be the instruction induced by the coordinate function \( f_i \). For any \( g \in A^n \), we may now find \( h \in S_f = \langle F^{(1)}, \ldots, F^{(n+2)} \rangle \) such that \( pr_n \circ g = h \circ pr_n \). Suppose that \( g = g^{(1)} \circ g^{(2)} \circ \cdots \circ g^{(\ell)} \), where \( g^{(k)} \in Y \). By grouping together the powers of \( I^{(j)} \), we may assume that \( g^{(k)} \in Y \cup \{(I^{(j)})^\lambda \colon 1 \leq \lambda \leq q - 1 \} \), so \( \ell \leq 3(q-1)nq^{n-1} + O(q^n) \). Denote \( \lambda = \sum_{i=1}^\rho \lambda_i 2^{i-1} \), with \( \lambda_i \in \{0,1\} \). Let \( h^{(0)} = F^{(n+1)} \), and for each \( 1 \leq k \leq \ell \), let

\[
h^{(k)} = \begin{cases} 
(F^{(n+2)})^\rho F^{(1)}F^{(n+1)} & \text{if } g^{(k)} = T^{(1)} \\
(F^{(n+2)})^\rho F^{(2)}F^{(n+1)} & \text{if } g^{(k)} = A^{(2)} \\
(F^{(j)})^{\lambda_1} F^{(n+2)}(F^{(j)})^{\lambda_2} \cdots F^{(n+2)}(F^{(j)})^{\lambda_\ell} F^{(n+1)} & \text{if } g^{(k)} = (I^{(j)})^\lambda.
\end{cases}
\]

Therefore, we may take \( h = h^{(0)} \circ h^{(1)} \circ \cdots \circ h^{(\ell)} \), which uses at most \( 2\rho \ell \) instructions from \( \{F^{(1)}, \ldots, F^{(n+2)}\} \). This shows that \( f \) is an n-universal transformation of size \( n + 2 \) and time \( 6\rho(q-1)nq^{n-1} + O(q^n) \).

\[ \square \]

Remark 1. For \( q = 3 \) or \( q = 5 \), there is a simpler n-universal transformation of size \( n + 2 \) whose time is strictly less than the time of the n-universal transformation \( f \) constructed in the proof of Theorem
Defining

\[(x)\tilde{f}_1 = \begin{cases} 
(x)pr[n] \circ i_1 & \text{if } x_{n+1} = x_{n+2} \\
(x)pr[n] \circ i_1 & \text{if } x_{n+1} \neq x_{n+2},
\end{cases}\]

\[(x)\tilde{f}_2 = \begin{cases} 
(x)pr[n] \circ i_2 & \text{if } x_{n+1} = x_{n+2} \\
(x)pr[n] \circ a_2 & \text{if } x_{n+1} \neq x_{n+2},
\end{cases}\]

\[(x)\tilde{f}_j = (x)pr[n] \circ i_j, \quad (3 \leq j \leq n)\]

\[(x)\tilde{f}_{n+1} = x_{n+2},\]

\[(x)\tilde{f}_{n+2} = \begin{cases} 
x_{n+2} & \text{if } x_{n+1} \neq x_{n+2} \\
x_{n+2} + 1 & \text{if } x_{n+1} = x_{n+2},
\end{cases}\]

we obtain an \(n\)-universal transformation \(\tilde{f}\) of size \(n+2\) and time \(t_{\tilde{f}}(n) = 3(q-1)q^n + O(q^n)\).

Observe that, for \(q = 7\) or \(q \geq 9\), we have \(t_{\tilde{f}}(n) > t_f(n)\), while, for \(q \in \{2, 4, 6, 8\}\), \(t_{\tilde{f}}(n) = t_f(n)\). However, for \(q = 3\) or \(q = 5\), we have \(t_{\tilde{f}}(n) < t_f(n)\); Table 1 compares explicitly the times of \(\tilde{f}\) and \(f\).

| \(q\) | \(t_f(n)\) | \(t_{\tilde{f}}(n)\) |
|------|-------------|------------------|
| 3    | \(6n3^n + O(3^n)\) | \(8n3^n + O(3^n)\) |
| 5    | \(60n5^{n-1} + O(5^n)\) | \(72n5^{n-1} + O(5^n)\) |

Table 1: Times of \(\tilde{f}\) and \(f\).

2.2 Universal transformations with minimal time

In the next theorem, we construct an \(n\)-universal transformation that, instead of having switches consisting of two registers (as in Example 2 or Theorem 3), it has a switch consisting of a one-error correcting code.

**Theorem 4.** For any \(n \geq 2\), there exists an \(n\)-universal transformation with time \(q^n + O(n)\).

**Proof.** We shall construct an \(n\)-universal transformation \(f\) with the required time. Foremost, we need the following notation. Let \(A = \mathbb{Z}_q\) and \(\rho := \lfloor \log_2 q \rfloor\). We then consider the following functions:

\[\text{odd} : A \rightarrow GF(2)\]

\[(a)\text{odd} = \begin{cases} 
1 & \text{if } a \equiv 1 \mod 2 \\
0 & \text{if } a \equiv 0 \mod 2,
\end{cases}\]

\[\text{err} : A \rightarrow A\]

\[(a)\text{err} = \begin{cases} 
 a + 1 & \text{if } a < q - 1 \\
 a - 1 & \text{if } a = q - 1.
\end{cases}\]

By applying them component-wise, we extend these functions to \(\text{odd} : A^k \rightarrow GF(2)^k\) and \(\text{err} : A^k \rightarrow A^k\), for any \(k \geq 1\). We also identify any \(v \in GF(2)^\rho\) with its lexicographic index \(\sum_{i=1}^\rho v_i 2^{i-1}\).
Moreover, let \( Q := q^n \) and \( n := Q + \lceil \log_2 Q \rceil + 1 \), and consider the \( (\hat{n}, Q, 3) \)-shortened Hamming code \( C \) in systematic form (see [13]). Let \( M \in \text{GF}(2)^{Q \times \hat{n}} \) be the generator matrix of \( C \), and, for \( H \subseteq [\hat{n}] \), denote by \( M_H \) the matrix formed with the \( H \)-th columns of \( M \). As \( C \) is a one-error correcting code, let

\[
\text{dec} : \text{GF}(2)^{\hat{n}} \rightarrow \{0, \ldots, \hat{n}\}
\]

\[
(v)_{\text{dec}} = \begin{cases} j & \text{if } v = c + e_j \text{ for some } c \in C \\ 0 & \text{otherwise}. \end{cases}
\]

Let \( (A^n)^A = \{\gamma^{(1)}, \ldots, \gamma^{(Q)}\} \) be the set of all coordinate functions \( A^n \rightarrow A \), and let \( r := \lceil ([\log_2 Q] + 1)/\rho \rceil \). The \( n \)-universal transformation \( f \in \text{Tran}(A^m) \), with \( m = 2n + Q + r \), is given as follows:

\[
(x)f_i = \begin{cases} \gamma \left( (x)_{\text{pr}[n]\setminus[n]} \right) \gamma \left( (x)_{\text{pr}[m]\setminus[2n]} \right) \text{odd dec} & \text{if } (x)_{\text{pr}[m]\setminus[2n]} \text{ odd dec} > Q, \\ (n + 1 \leq j \leq 2n), \\ (x)f_j = x_{j-n}, \\ (x)f_k = (x_k)\text{err}, \\ (x)f_l = \left( (x)_{\text{pr}[2n+Q]\setminus[2n]} \right) G_{[Q+\rho]\setminus[Q+(l-1)\rho]}, & \text{otherwise}, \\ 2n + 1 \leq k \leq 2n + Q, \\ (2n + Q + 1 \leq l \leq 2n + Q + r). \end{cases}
\]

The different parts of the transformation may be intuitively explained as follows:

- registers \( n + 1 \) to \( 2n \) maintain a copy of the original configuration of registers 1 to \( n \);
- instructions \( F^{(2n+1)}, \ldots, F^{(m)} \) are used to describe the coordinate function \( g_i \), where each register \( k \) from \( 2n + 1 \) to \( 2n + Q \) holds the bit \((x_k)\text{odd}\) while each register from \( 2n + Q + 1 \) to \( m \) holds \( \rho \) bits;
- registers 1 to \( n \) then have access to their original configuration and to the coordinate function they need to simulate.

The transformation \( g = (\gamma^{(k_1)}, \ldots, \gamma^{(k_n)}) \in \text{Tran}(A^n) \) is simulated as follows:

**Step 1.** Make a copy of the first \( n \) registers: \( F^{(n+1)} \circ \cdots \circ F^{(2n)} \).

**Step 2.** Encode \((x)_{\text{pr}[2n+Q]\setminus[2n]}\) into a codeword of \( C: F^{(2n+Q+1)} \circ \cdots \circ F^{(2n+Q+r)} \).

**Step 3.** For \( i \) from 1 to \( n \) do:

  **Step 3.1.** Add an error for \( \gamma^{(k_i)}: F^{(2n+k_i)} \).
  **Step 3.2.** Compute \( \gamma^{(k_i)}: F^{(i)} \).
  **Step 3.3.** Remove the error: \( F^{(2n+k_i)} \).

The time is then given by

\[
n + r + 3n = \frac{\log_2 Q}{\rho} + O(n) = q^n + O(n).
\]

**Conjecture 1.** Any \( n \)-universal transformation has time at least \( q^n \).

Provided Conjecture [11] holds, Theorem [11] indicates that the minimum time of an \( n \)-universal transformation is \( t(n) = q^n + O(n) \). As such, Theorem [3] indicates that we can achieve a time of \( O(t(n) \log t(n)) \) with only two additional registers.
3 Sequential simulation of transformations

Definition 5 (Sequential simulation). Let $m \geq n \geq 2$. We say that $f : A^m \rightarrow A^m$ sequentially simulates $g^{(1)}, \ldots, g^{(\ell)} \in \text{Tran}(A^n)$ if there exist $h^{(1)}, \ldots, h^{(\ell)} \in S_f \subseteq \text{Tran}(A^m)$ such that, for any $1 \leq i \leq \ell$,

$$\text{pr}_{[n]} \circ g^{(i)} = h^{(1)} \circ \cdots \circ h^{(i)} \circ \text{pr}_{[n]}.$$

The sequential time of simulation, denoted by $\text{st}_f(g^{(1)}, \ldots, g^{(\ell)})$, is the memoryless complexity of $h^{(1)} \circ \cdots \circ h^{(\ell)}$ with respect to $\{F^{(1)}, \ldots, F^{(m)}\}$.

Definition 6 (Complete $n$-universal). A transformation of $A^m$ is called complete $n$-universal if it may sequentially simulate any finite sequence in $\text{Tran}(A^n)$.

Define the maximal and minimal sequential times of $f$, denoted by $\text{st}_f^{\text{max}}$ and $\text{st}_f^{\text{min}}$, respectively, as follows:

$$\text{st}_f^{\text{max}} = \max \left\{ \text{st}_f(g^{(1)}, \ldots, g^{(q^n)}) : g^{(i)} \neq g^{(j)} \text{ for } i \neq j \right\},$$
$$\text{st}_f^{\text{min}} = \min \left\{ \text{st}_f(g^{(1)}, \ldots, g^{(q^n)}) : g^{(i)} \neq g^{(j)} \text{ for } i \neq j \right\}.$$

As the sequences considered in the above definitions must include each transformation in $\text{Tran}(A^n)$ exactly once, the relevant aspect when calculating maximal and minimal sequential times is the order in which the transformations of $A^n$ appear in the sequence.

Example 3. The $n$-universal transformation of Example 2 is in fact a complete $n$-universal transformation. Let $g^{(1)} = p^{(s_1)}, \ldots, g^{(\ell)} = p^{(s_{\ell})}$, then this sequence can be simulated as follows.

Step 1. Make a copy of the first $n$ registers: $F^{(n+1)} \circ \cdots \circ F^{(2n)}$.

Step 2. Turn all switches off: $F^{(2n+1)} \circ \cdots \circ F^{(2n+Q^n)}$.

Step 3. For $j$ from 1 to $\ell$ do:

- Step 3.1. Turn the switch on: $F^{(2n+Q^n+s_j)}$.
- Step 3.2. Compute $p^{(s_j)}$: $F^{(1)} \circ \cdots \circ F^{(n)}$.
- Step 3.3. Turn the switch off: $F^{(2n+s_j)}$.

The maximal sequential time of $f$ then satisfies

$$\text{st}_f^{\text{max}} \leq (n + 3 + o(1))Q^n.$$

By considering a sequence where $g^{(j-1)}$ and $g^{(j)}$ only differ by one coordinate function (i.e. a Gray code, which we shall use later on), Step 3.2 can be simplified by updating only the register corresponding to the differing coordinate function. The minimal sequential time of $f$ then satisfies

$$\text{st}_f^{\text{min}} \leq (4 + o(1))Q^n.$$

3.1 Complete universal transformations of minimal size

Theorem 5. The size of a complete $n$-universal transformation is at least $2n$.

Proof. Let $f$ be a complete $n$-universal transformation of size $m \geq n$. Identify the elements $x$ of $A^n$ with their lexicographic index $\sum_{i=1}^{n} x_ig^{(i)-1}$, and consider the sequence $g^{(0)}, g^{(1)}, \ldots, g^{(q^n-1)} \in \text{Tran}(A^n)$ defined by

$$(x)g^{(k)} = \begin{cases} 1 & \text{if } x = k, \\ 0 & \text{otherwise}. \end{cases}$$

By definition of complete $n$-universal, there is a sequence $h^{(0)}, h^{(1)}, \ldots, h^{(q^n-1)} \in S_f$ such that, for any $0 \leq i \leq q^n - 1$,

$$\text{pr}_{[n]} \circ g^{(i)} = h^{(0)} \circ \cdots \circ h^{(i)} \circ \text{pr}_{[n]}.$$
We claim that the rank of \( h^{(0)} : A^n \to A^n \) is at least \( q^n \). Indeed, if \( 0 \leq k_1, k_2 \leq q^n - 1 \) satisfy \((k_1)h^{(0)} = (k_2)h^{(0)}\), then
\[
(k_1)h^{(0)} \circ h^{(1)} \circ \cdots \circ h^{(k_1)} \circ \text{pr}_{[n]} = (k_2)h^{(0)} \circ h^{(1)} \circ \cdots \circ h^{(k_1)} \circ \text{pr}_{[n]},
\]
There exists \( k_1 = k_2 \), which means that all the states of \( A^n \) have distinct images under \( h^{(0)} \). Since \((A^n)g^{(0)} = \{0, 1\}\), and 0 is the unique state with image 1 under \( g^{(0)} \), we have
\[
\left|(A^n)h^{(0)} \circ \text{pr}_{[n]|[n]}\right| \geq q^n - 1.
\]
This implies that \((A^m)\text{pr}_{[n]|[n]}\) has at least \( q^n - 1 \) states, which is only possible when \( m \geq 2n \).

**Theorem 6.** Let \(|A| = q \geq 2\). Then, there exists a complete \( n \)-universal transformation \( \hat{f} \) of size \( m \) and \( \text{st}_f^{\text{max}}(n) \) as given in Table 2.

| \( m \) | \( \text{st}_f^{\text{max}}(n) \) |
|---|---|
| \( q = 2 \) | \( 2n + 3 \) | \( 3n2^{2n^2+n} + O(2^{2n^2+n}) \) |
| \( q = 3 \) or \( q = 5 \) | \( 2n + 2 \) | \( 6(q - 1)q^{n^2+n-1} + O(q^{n^2+n}) \) |
| \( q = 4 \) or \( q \geq 6 \) | \( 2n + 2 \) | \( 6\log_2(q)(q - 1)q^{n^2+n-1} + O(q^{n^2+n}) \) |

**Table 2:** Complete \( n \)-universal transformations.

**Proof.** Assume first that \( q = 4 \) or \( q \geq 6 \), and let \( \rho := \lceil \log_2 q \rceil \). Consider the \( n \)-universal transformation \( f \) of size \( n + 2 \) constructed in the proof of Theorem 3. Now, define the coordinate functions of \( \hat{f} \) by
\[
(x)\hat{f}_i = \begin{cases} x_{i+3} + i & \text{if } x_{i+1} - x_{i+2} = \rho + 1 \\ (x)\text{pr}_{[n|2]} \circ f_i & \text{otherwise}, \end{cases} \quad (1 \leq i \leq n),
\]
\[
(x)\hat{f}_{n+1} = x_{n+2},
\]
\[
(x)\hat{f}_{n+2} = x_{n+2} + 1,
\]
\[
(x)\hat{f}_j = x_{j-n-2}, \quad (n + 3 \leq j \leq 2n + 2).
\]
Intuitively, registers \( n + 3 \) to \( 2n + 2 \) maintain a copy of the original configuration of registers 1 to \( n \); again, registers \( n + 1 \) and \( n + 2 \) indicate which coordinate function to use but now the position \( \rho + 1 \) indicates that \( f_i, 1 \leq i \leq n \), must copy back the original values of the input from registers \( n + 3 \) to \( 2n + 2 \).

Let \( F^{(i)} \) and \( \hat{F}^{(i)} \) be the instructions induced by the coordinate functions \( f_i \) and \( \hat{f}_i \), respectively. Suppose that we want to sequentially simulate \( g^{(1)}, \ldots, g^{(\ell)} \in \text{Tran}(A^n) \). Since \( f \) is \( n \)-universal, there exist \( h^{(1)}, \ldots, h^{(\ell)} \in S_f = \{F^{(1)}, \ldots, F^{(n+2)}\} \) such that \( \text{pr}_{[n]} \circ g^{(i)} = h^{(i)} \circ \text{pr}_{[n]} \). For \( 1 \leq i \leq \ell \), define \( \hat{h}^{(i)} \in S_f \) by replacing every instruction \( F^{(k)} \) in \( h^{(i)} \) by \( \hat{F}^{(k)} \). Let
\[
C := \hat{F}^{(n+3)} \cdots \hat{F}^{(2n+2)} \quad \text{and} \quad B := (\hat{F}^{(n+2)})^{\rho+1}(\hat{F}^{(1)} \cdots \hat{F}^{(n)})\hat{F}^{(n+1)}.
\]
Then, for every \( 1 \leq i \leq \ell \), we have
\[
\text{pr}_{[n]} \circ g^{(1)} \circ \cdots \circ g^{(i)} = (\hat{F}^{(n+1)}C\hat{h}^{(1)})(B\hat{h}^{(2)})(B\hat{h}^{(3)}) \cdots (B\hat{h}^{(i)}) \circ \text{pr}_{[n]}.
\]
By Theorem 3, each $\hat{h}^{(i)}$ has memoryless complexity of at most $6\rho(q - 1)nq^{n^2} + O(q^n)$. Hence, sequences of length $\ell = q^{n^2}$ have maximal sequential time of $6\rho(q - 1)nq^{n^2} + O(q^{n^2} + n)$. For $q = 3$ or $q = 5$, let $\rho := 1$, and use the above construction of $\hat{f}$ with $\hat{f}$, as in Remark 1 instead of $f$.

The proof for $q = 2$ is very similar. The main difference is that, as the first and second coordinate functions must choose amongst three possibilities ($i_1$, $t_1$, or $x_{n+3}$, and $i_2$, $a_2$, or $x_{n+4}$, respectively), a switch consisting of two registers does not suffice; however, a switch of three registers is enough for our purposes. More formally, we now define the transformation $\hat{f} \in \text{Tran}(A^{2n+3})$ by

\[
(x)\hat{f}_i = \begin{cases} 
  x_{n+3+i} & \text{if } x_{n+2} \neq x_{n+3} \\
  (x)\text{pr}_{[n+2]} \circ f_i & \text{otherwise,}
\end{cases} (1 \leq i \leq n),
\]

\[
(x)\hat{f}_{n+1} = x_{n+2},
\]

\[
(x)\hat{f}_{n+2} = x_{n+2} + 1,
\]

\[
(x)\hat{f}_{n+3} = x_{n+2},
\]

\[
(x)\hat{f}_j = x_{j-n-3}, \quad (n + 4 \leq j \leq 2n + 3).
\]

Using a similar notation as above, define

\[
C := \hat{F}^{(n-4)} \cdots \hat{F}^{(2n+3)} \quad \text{and} \quad B := \hat{F}^{(n+3)}(\hat{F}^{(1)} \cdots \hat{F}^{(n)})(\hat{F}^{(n+3)}).
\]

For $1 \leq i \leq \ell$, define $\tilde{h}^{(i)} \in S_f$ by replacing every instruction $F^{(k)}$ in $h^{(i)}$ by $\hat{F}^{(k)}$ for $k \leq n + 1$ and by replacing every instruction $F^{(n+2)}$ in $h^{(i)}$ by $F^{(n+2)} \hat{F}^{(n+3)}$. Then, for every $1 \leq i \leq \ell$, we have

\[
\text{pr}_{[n]} \circ g^{(1)} \circ \cdots \circ g^{(i)} = (\hat{F}^{(n+3)}C\hat{h}^{(1)})(Bh^{(2)})(Bh^{(3)}) \cdots (Bh^{(i)}) \circ \text{pr}_{[n]}.
\]

The time analysis is similar as before. 

\[\square\]

### 3.2 Complete universal transformations with minimal sequential times

**Theorem 7.** Let $f$ be a complete $n$-universal transformation. Then, $q^{nq^n} \leq \text{st}^f_{\text{min}}$. Conversely, there exists a complete $n$-universal transformation $f$ such that $\text{st}^f_{\text{min}} = (1 + o(1))q^{nq^n}$.

Clearly, one always needs at least $q^{nq^n}$ updates to compute any sequence of length $q^{nq^n}$, so $q^{nq^n} \leq \text{st}^f_{\text{min}}$. In order to prove the upper bound in the theorem, we need several preliminary results about Gray codes.

As usual, let $|A| = q$ and $n \geq 2$. An $(n, q)$-Gray code is an ordering $(a^{(0)}, \ldots , a^{(q^n-1)})$ of the states in $A^n$ such that two consecutive states differ by only one coordinate: $d_H(a^{(i-1)} \mod q^n, a^{(i)}) = 1$ for all $0 \leq i \leq q^n - 1$, where $d_H$ is the Hamming distance. For any Gray code $G = (a^{(0)}, \ldots , a^{(q^n-1)})$, we denote the sequence $C(G) = (c^{(0)}, \ldots , c^{(q^n-1)}) \in [n]^{q^n}$ where $c^{(i)} \in [n]$ is the coordinate in which $a^{(i-1)} \mod q^n$ and $a^{(i)}$ differ. A run of length $\ell$ for $G$ is a sequence $c^{(i)}, \ldots , c^{(i+\ell-1)}$ of consecutive distinct elements of $C(G)$. We say that $G$ has $r(G)$ runs if $C(G)$ can be partitioned into $r(G)$ runs. For instance, the canonical $(n, 2)$-Gray code has $2^{n-1}$ runs. For $n = 2$, we have

\[
\begin{align*}
  a^{(0)} &= 00, \quad a^{(1)} = 01, \quad a^{(2)} = 11, \quad a^{(3)} = 10, \\
  c^{(0)} &= 1, \quad c^{(1)} = 2, \quad c^{(2)} = 1, \quad c^{(3)} = 2.
\end{align*}
\]

For $n = 3$, we have

\[
\begin{align*}
  a^{(0)} &= 000, \quad a^{(1)} = 001, \quad a^{(2)} = 011, \quad a^{(3)} = 010, \quad a^{(4)} = 110, \quad a^{(5)} = 111, \quad a^{(6)} = 101, \quad a^{(7)} = 100, \\
  c^{(0)} &= 1, \quad c^{(1)} = 3, \quad c^{(2)} = 2, \quad c^{(3)} = 3, \quad c^{(4)} = 1, \quad c^{(5)} = 3, \quad c^{(6)} = 2, \quad c^{(7)} = 3.
\end{align*}
\]
Clearly, any Gray code has at least \( q^n/n \) runs; we shall construct \((n,q)\)-Gray codes with \( o(q^n) \) runs for even \( q \).

**Lemma 1.** For any \( n \) a power of 2, there exists an \((n,2)\)-Gray code with \( o(2^n) \) runs.

**Proof.** We shall prove the result by induction on \( n \). The code \( G_2 \) is the canonical Gray code. Suppose \( G_n = (a^{(0)}, \ldots, a^{(2^n-1)}) \) (or simply written, 0 up to \( 2^n - 1 \)), then \( G_{2n} \) is given by

\[
\begin{align*}
(0,0), (0,1), (1,1), (1,2), \ldots, (2^n - 1, 2^n - 1), (2^n - 1, 0), \\
(2^n - 2, 0), (2^n - 2, 1), \ldots, (2^n - 3, 2^n - 1), (2^n - 3, 0), \\
\vdots \\
(2,0), (2,1), \ldots, (1,2^n - 1), (1,0) \end{align*}
\]

There are \( 2^{n-1} \) rows, each containing \( 2^{n+1} \) elements. The code \( G_4 \) is then

\[
\begin{align*}
G_4 &= \left(0000, 0001, 0101, 0111, 1111, 1110, 1010, 1000, \\
&1100, 1101, 1001, 1011, 0011, 0010, 0110, 0100\right), \\
C(G_4) &= (2,4,2,3,1,4,2,3,2,4,2,3,1,4,2,3),
\end{align*}
\]

which can be partitioned into six runs (instead of eight for the canonical code).

Let \( \Psi(n,d) \) denote the set of indices \( i \) such that the next occurrence of \( c^{(i)} \) appears at least \( d \) indices later. More formally, let \( \Gamma_n = (V,E) \) be the directed graph on \( V = \{0,\ldots,2^n - 1\} \) with arcs \( E = \{(i, i + 1 \mod 2^n - 1) : i \in V\} \) and let \( d(i,j) \) be the length of the path from \( i \) to \( j \) in \( \Gamma_n \), then

\[\Psi(n,d) = \{i : 0 < d(i,j) < d \Rightarrow c^{(j)} \neq c^{(i)}\}\]

For any \( d \), the Gray code \( G_n \) has at most

\[|\Psi(n,d)|/d + 2(2^n - |\Psi(n,d)|) + 1\]

runs. Indeed, split \( \Psi(n,d) \) into sequences \( s_1, \ldots, s_m \) of consecutive indices, where \( m \leq 2^n - |\Psi(n,d)|+1 \). Each sequence \( s_t \) of length \( l_t \) (\( 1 \leq t \leq m \)) can be partitioned into \( \lfloor l_t/d \rfloor \leq l_t/d + 1 \) runs, thus requiring at most \(|\Psi(n,d)|/d + 2^n - |\Psi(n,d)| + 1 \) runs to partition \( \Psi(n,d) \). Moreover, the indices outside of \( \Psi(n,d) \) can be partitioned into singleton runs; altogether, this yields \(|\Psi(n,d)|/d + 2(2^n - |\Psi(n,d)|)+1 \) runs.

Our strategy is then to find a distance \( d \) such that \( d = o(1) \) and \( 2^n - |\Psi(n,d)| = o(2^n) \). We have \(|\Psi(n,2)| = 2^n \) for all \( n \) and by construction,

\[|\Psi(2n,2d)| \geq 2^n|\Psi(n,d)| - 2^nd,\]

since the only ones that do not follow the simply doubling pattern are the ones at the end of every row. Denoting the largest power of two less than or equal to \( \log_2 n \) as \( l \), we then obtain

\[
\begin{align*}
|\Psi(n,l)| &\geq 2^{n/2}|\Psi(n/2,l/2)| - 2^{n/2}l \\
&\geq 2^{3n/4}|\Psi(n/4,l/4)| - l(2^{3n/4-1} + 2^n/2) \\
&\vdots \\
&\geq 2^{n-2n/l}|\Psi(2n/l,2)| - l(2^n-2n/l-l + \cdots + 2^n/2) \\
&= 2^n - o(2^n).\end{align*}
\]

\[\mathbb{Q.E.D.}\]

**Lemma 2.** For any \( n \), there exists an \((n,2)\)-Gray code with \( o(2^n) \) runs.
Proof. This is obtained by the usual “product” construction of Gray codes. Let \( m \) be the largest power of two less than or equal to \( n \). Denote the \((m,2)\)-Gray code from Lemma 3 by \((0,1,\ldots,2^m-1)\) and an \((n-m,2)\)-Gray code by \((0,1,\ldots,2^{n-m}-1)\). Now, construct an \((n,2)\)-Gray code as follows:

\[
\left( (0,0),\ldots,(0,2^n-1), \right.
\quad (1,0),\ldots,(1,2^m-1), \\
\vdots \\
\left. (2^{n-m}-1,0),\ldots,(2^{n-m}-1,2^m-1) \right) .
\]

This has at most \(2^{n-m} \cdot o(2^n) = o(2^n)\) runs.

Lemma 3. For any even \( q \) and any \( n \), there exists an \((n,q)\)-Gray code with \( o(q^n) \) runs.

Proof. Here again, we use a “product” construction, viewing each element of \([q]^n\) (= \(A^n\)) as an element of \([p]^n \times [2]^n\), where \( p = \frac{q}{2} \). We then combine any \((n,p)\)-Gray code with the \((n,2)\)-Gray code from Lemma 2 as follows:

\[
\left( (0,0),\ldots,(0,2^n-1), \right.
\quad (1,0),\ldots,(1,2^m-1), \\
\vdots \\
\left. (p^n-1,0),\ldots,(p^n-1,2^n-1) \right) .
\]

Clearly, this has at most \(p^n \cdot o(2^n) = o(q^n)\) runs.

For odd \( q \), we do not use a Gray code. Instead, an \((n,q)\)-pseudo-Gray code of length \( L \) is a sequence \( P = (p^{(0)},\ldots,p^{(L-1)}) \) of elements of \([q]^n\) such that every element of \([q]^n\) appears in the sequence and any two consecutive elements of the sequence only differ by one coordinate. (A pseudo-Gray code is a Gray code if every element appears exactly once.) Runs are defined for pseudo-Gray codes in the natural way and the number of runs is still denoted \( r(P) \); the redundancy \( R(P) \) of a pseudo-Gray code is \( R(P) = r(P) + L - q^n \).

Lemma 4. For any \( q \) and any \( n \), there exists an \((n,q^n)\)-pseudo-Gray code with redundancy \( o(q^n \cdot q^n) \).

Proof. If \( q \) is even, we use the Gray code from Lemma 3. Suppose that \( q \) is odd. Then \( Q := q^n \) is odd, so, again by Lemma 3, there is an \((n, Q - 1)\)-Gray code \( G \) with \( o(Q^n) \) runs. We shall construct an \((n,Q)\)-pseudo-Gray code by using \( G \) first, and then enumerating all of the remaining states in \([Q]^n\). It takes at most \( n \) steps to go from of these remaining states to another, and there are \( Q^n - (Q - 1)^n \) of them. Thus, the redundancy of this pseudo-Gray code is at most

\[
2n(Q^n - (Q - 1)^n) + o(Q^n) = 2n(q^nq^n - (q^n - 1)^n) + o(Q^n) \leq 2n \cdot nq^{(n-1)q^n} + o(Q^n) = o(Q^n).
\]

Finally, we may prove Theorem 4.

Proof of Theorem 4. We explicitly construct the complete \(n\)-universal transformation \( f \) of the statement of the theorem. Let \( Q := q^n \) and \( P = (p^{(0)},\ldots,p^{(L-1)}) \) be the \((n,Q)\)-pseudo Gray code of Lemma 4. We use the notation \( C(P) = (c^{(0)},\ldots,c^{(L-1)}) \), which is partitioned into \( r = r(P) \) runs \( R_1 = (c^{(0)},\ldots,c^{(\rho_1-1)}), \ldots, R_r = (c^{(\rho_r-1)},\ldots,c^{(L-1)}) \) and

\[
\tau : [r] \times [n] \to \{0,\ldots,Q^n - 1\} \\
(s,i) \tau = \begin{cases} 
\lambda & \text{if } \exists \lambda : i = c_\lambda \in R_s \\
0 & \text{if } i \notin R_s.
\end{cases}
\]
Let \( G = (\bar{a}(0), \ldots, \bar{a}(q^\sigma-1)) \) be a \((\sigma, q)\)-Gray code, where \( \sigma = \lceil \log_q r \rceil + 1 \). We use the index function

\[
\text{ind} : A^\sigma \to \{0, \ldots, q^\sigma - 1\}
\]

\[
(x) \text{ind} = j \text{ such that } \bar{a}_j = x
\]

and \( C(G) = (\bar{c}_0, \ldots, \bar{c}_{q^\sigma-1}) \). We denote the successor function for this code as

\[
S : A^\sigma \to A^\sigma
\]

\[
(x)S = \bar{a}_{(j+1) \mod(q^\sigma)} \quad (0 \leq j \leq q^\sigma - 1).
\]

Let \( m = 2n + \sigma + 2 \). Then the transformation \( f \) is defined as follows:

\[
(x)f_i = (x)\text{pr}_{[2n]\setminus[n]}p_k^{((x)\text{pr}_{[2n+\sigma]\setminus[2n]}\text{ind}, i)}r, \quad (1 \leq i \leq n),
\]

\[
(x)f_j = x_{j-n}, \quad (n + 1 \leq j \leq 2n),
\]

\[
(x)f_k = \begin{cases} (x)\text{pr}_{[2n+\sigma]\setminus[2n]}S_{k-2n} & \text{if } x_m = x_{m-1} \\ 1 & \text{if } x_m \neq x_{m-1}, \end{cases} \quad (2n + 1 \leq k \leq 2n + \sigma),
\]

\[
(x)f_{m-1} = x_m,
\]

\[
(x)f_m = x_m + 1.
\]

Intuitively, registers \( n + 1 \) to \( 2n \) maintain a copy of the original configuration of registers 1 to \( n \); registers \( 2n + 1 \) to \( 2n + \sigma \) form a counter indicating the run number in the pseudo-Gray code \( P \); registers \( m - 1 \) and \( m \) form a reset switch for the run counter.

The program computing \( P = (p^{(0)} = \text{id}, \ldots, p^{(L-1)}) \) in order goes as follows.

**Step 1.** Make a copy of the first \( n \) registers: \( F^{(n+1)} \circ \cdots \circ F^{(2n)} \).

**Step 2.** Reset switch on: \( F^{(m-1)} \).

**Step 3.** Reset run counter: \( F^{(2n+1)} \circ \cdots \circ F^{(2n+\sigma)} \).

**Step 4.** Reset switch off: \( F^{(m)} \).

**Step 5.** For \( s \) from 1 to \( r \) do:

**Step 5.1.** Compute \( p^{(r_{s-1})} \) to \( p^{(r_s-1)} \): \( F^{(r_{(s-1)})} \circ \cdots \circ F^{(r_{s-1})} \).

**Step 5.2.** Increment the run counter: \( F^{(2n+\bar{c}_{s+1})} \).

Total time:

\[
n + 1 + \sigma + 1 + \sum_{s=1}^{r} (r_s - r_{s-1} + 1) = L + r + O(\log r) = Q^n + o(Q^n).
\]

We finally prove that this transformation is complete. Let \( p^{(i_1)}, \ldots, p^{(i_\ell)} \in \text{Tran}(A^n) \) be a sequence of transformations. It is clear that applying Step 1 and then repeating \( \ell \) times Steps 2 to 5 will simulate \( \ell \) times the full sequence \( p^{(0)}, \ldots, p^{(L-1)} \). As such, \( p^{(i_1)} \) is simulated during the first iteration, \( p^{(i_2)} \) during the next, and so on until \( p^{(i_\ell)} \).

**Theorem 8.** Let \( f \) be a complete \( n \)-universal transformation. Then, \( nq^{\alpha n} \leq \text{st}_f^{\max} \). Conversely, there exists a complete \( n \)-universal transformation \( f \) such that \( \text{st}_f^{\max} = (n + 1 + o(1))q^{\alpha n} \).
Proof. Viewing any coordinate function $A^n \to A$ as an element in $\mathbb{Z}_Q$, with $Q = q^{nq} ≥ 4$, we give an ordering of $\text{Tran}(A^n) \cong \mathbb{Z}_Q^3$ such that any two consecutive transformations differ in all $n$ coordinate functions:

\[
\begin{align*}
(0, \ldots, 0), (1, \ldots, 1), \ldots, (Q - 1, \ldots, Q - 1), \\
(1, 0, \ldots, 0), (2, 1, \ldots, 1), \ldots, (0, Q - 1, \ldots, Q - 1), \\
\vdots \\
(0, 1, \ldots, 0), (1, 2, \ldots, 1), \ldots, (Q - 1, 0, \ldots, Q - 1), \\
\vdots \\
(Q - 1, \ldots, Q - 1, 0), (0, \ldots, 0, 1), \ldots, (Q - 2, \ldots, Q - 2, Q - 1).
\end{align*}
\]

Clearly, the time of simulation of such a sequence of transformations is at least $nq^{nq}$, so $nq^{nq} ≤ s_{\text{fmax}}$.

The construction of $f$ is similar to that of Theorem 4, it is still based on Hamming codes, except that it describes a whole sequence of transformation of $A^n$ at once. Let $\rho := \lfloor \log_2 q \rfloor$. We shall re-use the notation of odd and error introduced in Theorem 4. Moreover, for $\hat{\rho} := \rho^q$ and $\hat{n} := \hat{k} + \lfloor \log_2 \hat{k} \rfloor + 1$, consider the $(\hat{n}, \hat{k}, 3)$-shortened Hamming code $C$ in systematic form. Let $M \in \text{GF}(2)^{\hat{n} \times \hat{n}}$ be its generator matrix and, for $H \subseteq [\hat{n}]$, let $M_H$ be the matrix formed with the $H$-th columns of $M$. This is a one-error correcting code, so let

\[
\text{dec} : \text{GF}(2)^{\hat{n}} \to \{0, \ldots, \hat{n}\}
\]

\[
(v)_{\text{dec}} = \begin{cases} j & \text{if } v = c + e^j \text{ for some } c \in C \\ 0 & \text{otherwise.} \end{cases}
\]

We denote $(\text{Tran}(A^n))^Q = \left\{ \Gamma^{(1)}, \ldots, \Gamma^{(\hat{k})} \right\}$, where $\Gamma^{(j)} = (g^{j;0}, \ldots, g^{j;Q-1})$, and $r = \lceil \lceil \log_2 \hat{k} \rceil + 1 \rceil / \rho$. Let $m := 2n + \hat{k} + r + q^n + 2$. Let $(x)l \in \{0, \ldots, Q - 1\}$ be the index of $(x)pr_{[m-2]\backslash[2n+\hat{k}+r]}$ according to the $(q^n, q)$-Gray code $G = (a^{(0)}, \ldots, a^{(Q-1)})$, and let $C(G) = (c^{(0)}, \ldots, c^{(Q-1)})$. The index $(x)l$ shall work as an counter to decide which transformation will be simulated. For $x \in A^n$, define $(x)\tau := (x)pr_{[2n+\hat{k}+r]\backslash[2n]} \text{odd dec}$. The complete $n$-universal transformation $f$ is given as follows:

\[
(x)f_{i} = \begin{cases} x_i, & \text{if } (x)\tau > \hat{k}, \\ (x)pr_{[2n]\backslash[n]} \circ g_i((x)\tau; (x)l), & \text{otherwise}, \end{cases}
\]

\[
(1 \leq i \leq n),
\]

\[
(x)f_{j} = x_{j-n}, \quad (n+1 \leq j \leq 2n),
\]

\[
(x)f_{k} = (x)\text{err}, \quad (2n+1 \leq k \leq 2n + \hat{k}),
\]

\[
(x)f_{l} = (x)pr_{[2n+\hat{k}\backslash[2n]} \text{ odd})M_{[k+l\rho]\backslash[k+(l-1)\rho]}, \quad (2n + \hat{k} + 1 \leq l \leq 2n + \hat{k} + r),
\]

\[
(x)f_{s} = \begin{cases} a_s((x)l+1 \mod(Q)), & \text{if } x_m = x_{m-1}, \\ 0, & \text{if } x_m \neq x_{m-1}, \end{cases}
\]

\[
(2n + \hat{k} + r + 1 \leq s \leq m - 2),
\]

\[
(x)f_{m-1} = x_{m},
\]

\[
(x)f_{m} = x_{m} + 1.
\]

Let $\lambda \geq 1$ and $\Lambda = Q\lambda$. The sequence $g^{(i_1;0)}, \ldots, g^{(i_1;Q-1)}, \ldots, g^{(i_3;0)}, \ldots, g^{(i_3;Q-1)}$ of length $\Lambda$ is simulated as follows:
Step 1. Make a copy of the first \( n \) registers: \( F^{(n+1)} \circ \cdots \circ F^{(2n)} \).

Step 2. Encode \( (x) \text{pr}_{[2n+k]\setminus[2n]} \) into a codeword of \( C \): \( F^{(2n+k+1)} \circ \cdots \circ F^{(2n+k+r)} \).

Step 3. Reset the counter \( (x)l \): \( F^{(m-1)} \circ F^{(m)} \circ F^{(2n+k+r+1)} \circ \cdots \circ F^{(m-2)} \circ F^{(m-1)} \).

Step 4. For \( \mu \) from 1 to \( \lambda \) do:

Step 4.1. Add an error to \( \Gamma^{(i_\mu)} = (g^{(i_\mu;0)}, \ldots, g^{(i_\mu;Q-1)}) \): \( F^{(2n+i_\mu)} \).

Step 4.2. For \( \sigma \) from 0 to \( Q-1 \) do:

Step 4.2.1. Compute \( g^{(i_\mu;\sigma)} : F^{(1)} \circ \cdots \circ F^{(n)} \).

Step 4.2.2. Increment the counter \( (x)\ell \) according to the Gray code: \( F^{(2n+k+r+c(\sigma+1))} \).

Step 4.3. Remove the error: \( F^{(2n+i_\mu)} \).

For \( \Lambda = Q^n \), the time to simulate this sequence is then given by
\[
n + r + (q^n + 3) + Q^{n-1}(Q(n + 1) + 2) = (n + 1 + o(1))Q^n.
\]

Note that the complete \( n \)-universal transformation \( \hat{f} \) of size \( 2n + 2 \) (or \( 2n + 3 \) when \( q = 2 \)) given in Theorem 6 does not have a very high sequential time compared with complete \( n \)-universal transformations of minimal sequential time; indeed, we may see that \( \text{st}_{f_{\text{max}}} \) is equal \( O(\text{st}_{f_{\text{max}}} \log \text{st}_{f_{\text{max}}}) \), with \( f \) as in Theorem 8.

4 Simulation of transformations in parallel

So far, we have looked at sequential updates (i.e. one register at a time). This is a strong constraint for MC: if we were allowed to update all registers at once, then any function could be computed in only one time step. However, in our model of universal simulation, this is actually a strength and a necessity.

We extend our framework to consider the following type of simulations.

**Definition 7** (Parallel simulation). Let \( m \geq n \geq 1 \). We say that \( f : A^m \to A^m \) simulates in parallel \( g : A^n \to A^n \) if there exists \( h \in \{ f \} \) such that \( \text{pr}_{[n]} \circ g = h \circ \text{pr}_{[n]} \).

We also consider the slightest form of asynchrony in sequential simulations. For \( f \in A^m \to A^m \), define \( F^{(m)} : A^m \to A^m \) by
\[
(x)F^{(m)} := ((x)f_1, (x)f_2, \ldots, (x)f_{m-1}, x_m).
\]

**Definition 8** (Quasi-parallel sequential simulation). Let \( m \geq n \geq 1 \). We say that \( f : A^m \to A^m \) sequentially simulates in quasi-parallel \( g^{(1)}, \ldots, g^{(\ell)} \in \text{Tran}(A^n) \) if there exist \( h^{(1)}, \ldots, h^{(\ell)} \in \langle F^{(m)} \rangle \) such that, for all \( 1 \leq i \leq \ell \),
\[
\text{pr}_{[n]} \circ g^{(i)} = h^{(1)} \circ \cdots \circ h^{(i)} \circ \text{pr}_{[n]},
\]
and the instruction \( F^{(m)} \) appears at most once in the program \( h^{(1)} \circ \cdots \circ h^{(\ell)} \).

Say that \( f : A^m \to A^m \) is a complete quasi-parallel \( n \)-universal transformation if it may sequentially simulate in quasi-parallel any finite sequence in \( \text{Tran}(A^n) \).

**Theorem 9.** For any \( m \geq n \geq 1 \), there is no transformation \( f : A^m \to A^m \) that may simulate in parallel every transformation in \( \text{Tran}(A^n) \). However, for any \( n \geq 1 \), there exists a complete quasi-parallel \( n \)-universal transformation.
Proof. Suppose that \( f : A^m \to A^m \) may simulate in parallel every transformation in \( \text{Tran}(A^n) \). For \( a, b \in A^n, a \neq b, \) consider the constant transformations \( g^{(a)}, g^{(b)} \in \text{Tran}(A^n) \) defined by \( (x)g^{(a)} = a \) and \( (x)g^{(b)} = b, \) for all \( x \in A^n \). Then, by definition of parallel simulation, there exist integers \( k_a < k_b \) such that, for all \( x \in A^n, \)

\[
(x)f^{k_a} \circ pr_{[n]} = (x)pr_{[n]} \circ g^{(a)} = a \quad \text{and} \quad (x)f^{k_b} \circ pr_{[n]} = (x)pr_{[n]} \circ g^{(b)} = b.
\]

But now we obtain a contradiction:

\[
b = (x)f^{k_b} \circ pr_{[n]} = (x)f^{k_b-k_a} \circ f^{k_a} \circ pr_{[n]} = a.
\]

Let \( |A| = q, Q = q^n \) and denote \( \text{Tran}(A^n) = \{p^{(2)}, \ldots , p^{(Q^n+1)} = \text{id}\}; \) moreover, let \( p^{(0)}, p^{(1)}, \) and \( p^{(c)} \) for \( c \geq Q^n + 2 \) all be equal to the identity. Let \( m = (Q^n + 1)n + q^n + 3. \) Now we shall construct a complete quasi-parallel \( n \)-universal transformation \( f. \) Let \( C \in \{0, \ldots , Q^n-1\} \) be the lexicographic index of \( (x)pr_{[(Q^n+2)n+q^n+1]\setminus[(Q^n+2)n]} \). We further denote \( f_C := (f_{(Q^n+2)n+1}, \ldots , f_{(Q^n+2)n+q^n+1}) \). Define \( f \) by:

\[
(x)f_i = (x)pr_{[(C+1)n]\setminus(Cn)}P_i^{(C)}, \quad (1 \leq i \leq n),
\]

\[
(x)f_j = x_{j-n}, \quad (n+1 \leq j \leq (Q^n+2)n),
\]

\[
(x)f_C = \begin{cases} 
(C + 1) \mod (Q^n + 2) & \text{if } x_m = x_{m-1} \\
2 & \text{if } x_m \neq x_{m-1},
\end{cases}
\]

\[
(x)f_{m-1} = x_m,
\]

\[
(x)f_m = x_m + 1.
\]

Let \( F^{[(m-1)]} := (f_1, \ldots , f_{m-1}). \) We prove that \( f \) is a complete quasi-parallel \( n \)-universal transformation by giving a program simulating the sequence \( p^{(2)}, \ldots , p^{(Q^n+1)} \) repeated \( \ell \) times for any \( \ell \geq 1. \)

**Step 1.** Do \( F^{[(m-1)]} \), Note that this makes \( C \) equal to \( C + 1 \mod (Q^n + 2) \) or 2, and turns the switch off: \( x_{m-1} = x_m. \)

**Step 2.** Do \( F^{[m]} \). This turns the switch on: \( x_m \neq x_{m-1}. \)

**Step 3.** Do \( F^{[(m-1)]} \). This makes \( C = 2 \), and turns the switch on \( x_{m-1} = x_m. \) Note that the contents of the first \( n \) registers are now contained in \( (x)pr_{[3n]\setminus[2n]} \).

**Step 4.** Do \( (F^{[(m-1)]})^Q \). At each iteration, this increases the counter \( C = (C + 1) \mod (Q^n + 2) \),

and so it computes the whole sequence \( p^{(2)}, \ldots , p^{(Q^n+1)}. \) Note that the original value of \( (x)pr_{[n]} \) is back in \( (x)pr_{[n]} \).

**Step 5.** Do \( (F^{[(m-1)]})^{(\ell-1)(Q^n+2)} \) in order to compute \( \ell - 1 \) iterations of \( p^{(0)}, \ldots , p^{(Q^n+1)}. \)

Now, as in the proof of Theorem \( \Box \) in order to simulate an arbitrary sequence \( p^{(i_1)}, \ldots , p^{(i_\ell)} \in \text{Tran}(A^n) \), we use the program above to simulate \( \ell \) times the full sequence \( p^{(2)}, \ldots , p^{(Q^n+1)} \).

5 Acknowledgment

We would like to thank Adrien Richard and Jean-Paul Comet for stimulating discussions. This work was supported by the EPSRC grant EP/K033956/1.
References

[1] R. Ahlswede, N. Cai, S.-Y.R. Li, and R.W. Yeung, Network information flow, IEEE Trans. Inform. Theory 45 (2000), no. 4, 1204–1216.

[2] S. Burckel, Closed iterative calculus, Theoret. Comput. Sci. 158 (1996), 371–378.

[3] ______, Elementary decompositions of arbitrary maps over finite sets, J. Symbolic Comput. 37 (2004), no. 3, 305–310.

[4] S. Burckel, E. Gioan, and E. Thomé, Mapping computation with no memory, Proc. International Conference on Unconventional Computation (Ponta Delgada, Portugal), 2009, pp. 85–97.

[5] ______, Computation with no memory, and rearrangeable multicast networks, Disc. Math. Theor. Comput. Sci. 16 (2014), 121–142.

[6] S. Burckel and M. Morillon, Three generators for minimal writing-space computations, Theor. Inform. Appl. 34 (2000), 131–138.

[7] ______, Quadratic sequential computations of boolean mappings, Theory Comput. Sys. 37 (2004), no. 4, 519–525.

[8] ______, Sequential computation of linear boolean mappings, Theoret. Comput. Sci. 314 (2004), 287–292.

[9] P. J. Cameron, B. Fairbairn, and M. Gadouleau, Computing in matrix groups without memory, Chicago J. Theoret. Comput. Sci. (2014), no. 08, 1–16.

[10] ______, Computing in permutation groups without memory, Chicago J. Theoret. Comput. Sci. (2014), no. 07, 1–20.

[11] P. Dömösi and C. L. Nehaniv, Algebraic theory of automata networks: An introduction, SIAM Monographs on Discrete Mathematics and Applications, 2005.

[12] M. Gadouleau and S. Riis, Memoryless computation: New results, constructions, and extensions, Theoret. Comput. Sci. 562 (2015), 129–145.

[13] J. Kari, Theory of cellular automata: A survey, Theoret. Comput. Sci. 334 (2005), 3–33.

[14] F. J. MacWilliams and N. J. A. Sloane, The theory of error-correcting codes, North-Holland, Amsterdam, 1977.

[15] B. Martin, A universal cellular automaton in quasi-linear time and its s–m–n form, Theoret. Comput. Sci. 123 (1994), 199–237.