Linearization of Automatic Arrays and Weave Specifications

David Sprunger

Indiana University

June 24, 2013
The original intention was to see how to extend the work

Automatic sequences and zip-specifications.

from zip-specifications of automatic sequences to automatic arrays.
Outline:

1. Background
2. The connection between sequences and arrays
3. Consequences
Let $\Delta$ be a finite set.

A **sequence in** $\Delta$ is a map $\mathbb{N} \to \Delta$ which we will usually represent like

$$\sigma = \sigma_0\sigma_1\sigma_2\sigma_3\sigma_4\sigma_5\sigma_6\sigma_7 \ldots$$

An **array in** $\Delta$ is a map $\mathbb{N} \times \mathbb{N} \to \Delta$ which we will usually represent like

$$a = \begin{array}{cccc}
& & & \\
\vdots & a_{20} & a_{21} & a_{22} & \ldots \\
\vdots & a_{10} & a_{11} & a_{12} & \ldots \\
a_{00} & a_{01} & a_{02} & \ldots \\
& & & \\
\end{array}$$

We use the functor $F_k : \text{Set} \to \text{Set}$ which acts on objects by $F_kX = \Delta \times X^k$ and on morphisms by $F_kf = \text{id}_\Delta \times f$. 
Sequences as coalgebras

Let \( S = \{ \sigma \in \Delta^\omega \} \) be the set of infinite sequences of symbols from \( \Delta \). We can make it into an \( F_k \)-coalgebra as follows.

The **sequence projection** \( \pi_{\frac{i}{k}} : S \rightarrow S \) is defined by

\[
\pi_{\frac{i}{k}}(\sigma) = \pi_{\frac{i}{k}}(\sigma_0 \sigma_1 \sigma_2 \sigma_3 \ldots) = \sigma_i \sigma_{i+k} \sigma_{i+2k} \sigma_{i+3k} \ldots
\]

(More standardly, \( \pi_{\frac{i}{k}} \) is written as \( \pi_{i,k} \).)

The **head function** \( \text{hd} : S \rightarrow \Delta \) is defined by

\[
\text{hd}(\sigma) = \sigma_0
\]

The \( F_k \)-coalgebra structure on \( S \) is given by

\[
\langle \text{hd}, \pi_{\frac{0}{k}}, \pi_{\frac{1}{k}}, \ldots, \pi_{\frac{k-1}{k}} \rangle : S \rightarrow \Delta \times S^k.
\]
Theorem (Kupke & Rutten and [6])

This coalgebra is final for the subcategory of $F_k$-coalgebras $(C, \langle o, c_0, c_1, \ldots, c_{k-1} \rangle)$ which share the property that $o \circ c_0 = o$. (Called zero-consistent coalgebras)

Roughly, the reason for finality is that the sequence projection maps give nearly unique addresses for each of the entries:

$$\sigma_0 \sigma_1 \sigma_2 \sigma_3 \sigma_4 \sigma_5 \sigma_6 \sigma_7 \sigma_8 \sigma_9 \sigma_{10} \sigma_{11} \sigma_{12} \sigma_{13} \sigma_{14} \sigma_{15} \sigma_{16} \ldots$$

$$\pi_0^\frac{1}{2} \quad \pi_1^\frac{1}{2} \quad \text{hd}$$

To get the entry $\sigma_5$ we must apply $\pi_1^\frac{1}{2}$, then $\pi_0^\frac{1}{2}$, then finally $\pi_1^\frac{1}{2}$ and hd.

$$\sigma_0 \sigma_1 \sigma_2 \sigma_3 \sigma_4 \sigma_5 \sigma_6 \sigma_7 \sigma_8 \sigma_9 \sigma_{10} \sigma_{11} \sigma_{12} \sigma_{13} \sigma_{14} \sigma_{15} \sigma_{16} \ldots$$
A **deterministic finite automaton with output (DFAO)** looks like this:

![DFAO Diagram](image)

You get a single output symbol by running the machine on an input string and seeing which state it ends up in:

- **Input:** 101  |  **Output:** L

A sequence $\sigma = \sigma_0\sigma_1\sigma_2 \ldots$ is **$k$-automatic** if there is a $k$-DFAO so that $\sigma_n$ is the output of the DFAO when given the input $[n]_k$. The sequence generated by the above DFAO is called the Thue-Morse sequence:

$$LRRLRLLRRLRLLRLLRRLRLLRLLR...$$
DFAOs as $F$-coalgebras

We can make the previous DFAO into an $F_2$-coalgebra using its set of states, $Q$, along with the transition and final output maps, $\delta$ and $f$ respectively:

$$Q \langle f, \delta(\cdot, 0), \delta(\cdot, 1) \rangle \rightarrow \Delta \times Q \times Q$$

By finality we immediately get a map, seq, into sequences in $\Delta$

$$Q \langle f, \delta(\cdot, 0), \delta(\cdot, 1) \rangle \rightarrow \Delta \times Q \times Q$$

seq

$$S \langle \text{hd}, \pi_0^{\frac{1}{2}}, \pi_1^{\frac{1}{2}} \rangle \rightarrow \Delta \times S \times S$$

The seq map takes a state $q \in Q$ to the sequence the DFAO generates when using that state as the start state.
### An array coalgebra

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

\[ \mathbf{c} : A \rightarrow \Delta \]

\[ \pi_{0,2}^{0,1} : A \rightarrow A \]

\[ \pi_{0,2}^{1,1} : A \rightarrow A \]

\[ \pi_{0,2}^{1,1} : A \rightarrow A \]

\[ \pi_{1,2}^{0,1} : A \rightarrow A \]
Proposition

The array coalgebra \((A, \langle c, \pi_{k, l} \rangle)\) is final for the zero-consistent coalgebras of \(F_{kl}X = \Delta \times X^{kl}\).

Recall the sequence coalgebra \((S, \langle \text{hd}, \pi_{k,l} \rangle)\) is also final for this functor. This means

Corollary

There is an isomorphism of coalgebras, \(\varphi_{k,l}\), as shown:

\[
\begin{array}{ccc}
S & \xrightarrow{\langle \text{hd}, \pi_{k,l} \rangle} & \Delta \times S^{kl} \\
\varphi & & F\varphi \\
A & \xrightarrow{\langle c, \pi_{k,l} \rangle} & \Delta \times A^{kl}
\end{array}
\]
What does $\varphi_{k,l}$ look like?

We choose a correspondence between sequence and array projections. For simplicity, we’ll take $k = l = 2$ and choose

<table>
<thead>
<tr>
<th>$S$</th>
<th>$A$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$hd$</td>
<td>$c$</td>
</tr>
<tr>
<td>$\pi_{0}\frac{4}{4}$</td>
<td>$\pi_{0}\frac{2}{2}$</td>
</tr>
<tr>
<td>$\pi_{1}\frac{4}{4}$</td>
<td>$\pi_{0}\frac{1}{2}$</td>
</tr>
<tr>
<td>$\pi_{2}\frac{4}{4}$</td>
<td>$\pi_{1}\frac{0}{2}$</td>
</tr>
<tr>
<td>$\pi_{3}\frac{4}{4}$</td>
<td>$\pi_{1}\frac{1}{2}$</td>
</tr>
</tbody>
</table>

So, $\sigma_5 = hd(\pi_{1}\frac{1}{4}(\pi_{1}\frac{1}{4}(\sigma)))$ corresponds to $c(\pi_{0}\frac{1}{2}(\pi_{0}\frac{1}{2}(\varphi_{2,2}(\sigma))))$. 
What does $\varphi_{2,2}$ look like?

We refer to $\varphi_{k,l} : S \rightarrow A$ as “ord” and $\varphi_{k,l}^{-1} : A \rightarrow S$ as “lin”.

David Sprung (Indiana University)
$k, l$-automatic arrays

A $k, l$-DFAO is like a DFAO except we make the additional assumption that its input alphabet consists of pairs of digits from base $k$ and base $l$. An example:

We determine the entry in the array at position $(m, n)$ by pairing the digits of $[m]_k$ and $[n]_l$ and running the DFAO on those pairs.
### Fractals: Sierpinski gasket

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 1 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 1
0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1 1
0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 1 0 1
0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 1 0 1 0 1
0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1
0 0 0 0 0 1 1 0 0 0 0 1 1 1 1 0 0 1 1 0 0 1 1
0 0 0 0 1 1 0 0 0 0 1 0 1 0 1 0 0 1 0 0 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```
Fractals: Sierpinski carpet

\[
\begin{array}{cccccccccccc}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\end{array}
\]
Uniting dozens of seemingly disparate results from different fields, this book combines concepts from mathematics and computer science to present the first integrated treatment of sequences generated by “finite automata.”

The authors apply the theory to the study of automatic sequences and their generalizations, such as Sturmian words and SLLS-regular sequences. And further, they provide applications to number theory (particularly to formal power series and transcendence in finite characteristic), physics, computer graphics, and music.

Starting from first principles wherever feasible, basic results from combinatorics on words, numeration systems, and models of computation are discussed. Thus this book is suitable for graduate students or advanced undergraduates, as well as for mature researchers wishing to know more about this fascinating subject.

Jean-Paul Allouche is Directeur de Recherche at CNRS, LRI, Orsay. He has written some 90 papers in number theory and combinatorics on words. He is on the editorial board of Advances in Applied Mathematics and on the scientific committee of the Journal de Théorie des Nombres de Bordeaux.

Jeffrey Shallit is Professor of Computer Science at the University of Waterloo. He has written 80 articles on number theory, algorithms, formal languages, combinatorics on words, computer graphics, history of mathematics, algebra and automata theory. He is the editor-in-chief of the Journal of Integer Sequences and co-author of Algorithmic Number Theory.
lin preserves automaticity

**Theorem (Cobham and Allouche & Shallit)**

A sequence $\sigma$ is $kl$-automatic $\iff$ The $kl$-kernel of $\sigma$ is finite

An array $a$ is $k, l$-automatic $\iff$ The $k, l$-kernel of $a$ is finite

**Theorem**

An array $a$ is $k, l$-automatic iff $\text{lin}(a)$ is $kl$-automatic. Similarly, a sequence $\sigma$ is $kl$-automatic iff $\text{ord}(\sigma)$ is $k, l$-automatic.
Zip specifications

Automatic sequences have compact descriptions in terms of zipping other sequences. For example, the Thue-Morse sequence from earlier

\[
\text{LRRLRLLRLLRLLRLLRLLRLLRLLRLLRLLRLLRLLRLLRLLRLLR...}
\]

can be described with the “zip equations”

\[
\begin{aligned}
    x &= \text{zip}(x, y), & \text{hd}(x) &= L \\
    y &= \text{zip}(y, x), & \text{hd}(y) &= R
\end{aligned}
\]

\[
\begin{aligned}
    x^* &= LRRLRLRR \\
    y^* &= RLLRRLRL
\end{aligned}
\]

Theorem (Grabmayer, Endrullis, Hendriks, Klop & Moss)

A sequence is \( k \)-automatic iff it has a \( \text{zip}_k \) specification. Moreover, it is decidable whether two zip specifications generate the same sequence.
We can describe this array with a system of equations and corner points, as below:

\[
\begin{align*}
\begin{bmatrix}
1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\
0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\
1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\
1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 
\end{bmatrix}
\end{align*}
\]

\[
x = \mathbf{wv} \begin{pmatrix} y & x \\ x & y \end{pmatrix} \quad \text{and} \quad c(x) = 0
\]

\[
y = \mathbf{wv} \begin{pmatrix} x & y \\ y & x \end{pmatrix} \quad \text{and} \quad c(y) = 1
\]
We can design a reasonable system of weave specifications where every weave term is the z-ordering of a zip term. Conversely, the z-ordering of a zip specification yields a weave specification.
The array $a$ is $k, l$-automatic \( \iff \) $a$ has a weave specification

The sequence $\text{lin}(a)$ is $kl$-automatic \( \iff \) $\text{lin}(a)$ has a zip specification

1. $\text{lin}$ preserves automaticity
2. Weave specifications are z-orderings of zip specifications
3. Grabmayer, Endrullis, Hendriks, Klop and Moss

**Theorem**

An array is $k, l$-automatic iff it has a $\text{wv}_{k,l}$ specification. Moreover, it is decidable whether two weave specifications have the same solution.
Further research

1. This suggests a method for investigating automatic arrays.
2. Higher dimensions?
3. (Partial result) Progress toward zip specifications of mixed arity?