Morphic words and recursion schemes

Laurent BRAUD

FREC meeting, LABRI

6 dec 2011
Graphs and trees with decidable MSO-theory:

- pushdown/Caucal hierarchy
- recursion schemes: [Damm], and recently [Knapik-Niwiński-Urzyczyn], [Ong]
- infinite words: ult. periodic, morphic [Carton-Thomas]
Context

Graphs and trees with decidable MSO-theory:
  ▶ pushdown/Caucal hierarchy
  ▶ recursion schemes: [Damm], and recently [Knapik-Niwiński-Urzyczyn], [Ong]
  ▶ infinite words: ult. periodic, morphic [Carton-Thomas]

This talk:
1. relationship between order-1 schemes and morphic words
2. extension to order 2
Recursion schemes: first order

Term grammar with

- terminals $T = \{a, b, f, g \ldots\}$,
- nonterminals $N = \{S, F, G \ldots\}$,
- a specific starting nonterminal $S$
- one rewriting rule per nonterminal, using variables $\mathcal{X} = \{x, y \ldots\}$.

Every symbol $\alpha$ has fixed arity $\rho(\alpha)$. 
Recursion schemes : first order

Term grammar with

- terminals $T = \{a, b, f, g \ldots \}$,
- nonterminals $N = \{S, F, G \ldots \}$,
- a specific starting nonterminal $S$
- one rewriting rule per nonterminal, using variables $\mathcal{X} = \{x, y \ldots \}$.

Every symbol $\alpha$ has fixed arity $\rho(\alpha)$.

$$\begin{align*}
T &= \{f, g, a\} \\
N &= \{S, F\}
\end{align*}$$

Here $\rho(f) = 3, \rho(F) = \rho(g) = 1, \rho(a) = 0.$
Recursion schemes: first order

\[ S \rightarrow F \quad F \rightarrow f \]
\[ F \rightarrow F \]
\[ F \rightarrow F \]
\[ F \rightarrow F \}

A scheme builds a (possibly) infinite tree of terminals.

\[ S \Rightarrow F \]
\[ F \quad F \]
\[ F \quad F \]
\[ F \quad F \]
\[ F \quad F \]

\[ a \]
Recursion schemes: first order

A scheme builds a (possibly) infinite tree of terminals.
Recursion schemes: first order

A scheme builds a (possibly) infinite tree of terminals.
Recursion schemes: first order

A scheme builds a (possibly) infinite tree of terminals.

\[ S \Rightarrow F \]
\[ F \Rightarrow a \]
\[ F \Rightarrow x \]
\[ f \]
\[ F \]
\[ g \]
\[ x \]
\[ a \]

\[ S \Rightarrow^3 f \]
\[ f \]
\[ a \]
\[ F \]
\[ F \]
\[ g \]
\[ F \]
\[ a \]
\[ g \]
\[ a \]
\[ g \]
\[ a \]
\[ a \]
Recursion schemes : first order

A scheme builds a (possibly) infinite tree of terminals.
We are interested in infinite words that appear in schemes.
Frontiers of limit trees

We are interested in infinite words that appear in schemes.

\[ S \implies F \quad \quad F \implies x \quad \implies x \quad \implies x \quad \implies f \quad \implies f \quad \implies f \quad \implies F \]

\[ [S] = \]

\[ \text{Fr}([S]) = abaab \ldots a^{2^i}b \ldots \]
Let $T$ be an infinite term and let the *frontier* $\text{Fr}(T)$ be the colored order of leaves in left-right order.
Let $T$ be an infinite term and let the *frontier* $\text{Fr}(T)$ be the colored order of leaves in left-right order. The *$\omega$-frontier* $\omega\text{-Fr}(T)$ is the initial part of $\text{Fr}(T)$ of type $\omega$, when it exists.
Frontiers of limit trees

Let $T$ be an infinite term and let the frontier $\text{Fr}(T)$ be the colored order of leaves in left-right order. The $\omega$-frontier $\omega\text{-Fr}(T)$ is the initial part of $\text{Fr}(T)$ of type $\omega$, when it exists.

**Proposition**

For any tree generated by an order-1 scheme, there is a tree generated by a order-1 scheme where the rightmost branch is the only infinite branch, and with the same $\omega$-frontier.

The trees with one infinite rightmost branch are called *combs*. 
Morphic words

$\Sigma$ is an alphabet. A *morphism* $\tau$ on $\Sigma^*$ is such that

$$\tau(ab) = \tau(a)\tau(b).$$
Morphic words

Σ is an alphabet. A morphism τ on Σ* is such that

\[ \tau(ab) = \tau(a)\tau(b). \]

Let τ be a morphism on Σ* s.t. there is \( a \in \Sigma \) with \( \tau(a) \in a\Sigma \).

\[
\begin{align*}
\tau(a) &= au \\
\tau^2(a) &= au\tau(u) \\
\tau^3(a) &= au\tau(u)\tau^2(u) \\
\cdots \\
\tau^\omega(a) &= au\tau(u)\cdots
\end{align*}
\]

Words \( \sigma(\tau^\omega(a)) \) are morphic words, where \( \sigma \) is another morphism.
Morphic words: example

\[ \tau(a) = abcc \quad \sigma(a) = a \]
\[ \tau(b) = b \quad \sigma(b) = b \]
\[ \tau(c) = cc \quad \sigma(c) = a \]
Morphic words : example

\[
\begin{align*}
\tau(a) &= abcc & \sigma(a) &= a \\
\tau(b) &= b & \sigma(b) &= b \\
\tau(c) &= cc & \sigma(c) &= a
\end{align*}
\]

\[
\tau(a) = abcc
\]
Morphic words: example

\[\tau(a) = abcc\quad\sigma(a) = a\]
\[\tau(b) = b\quad\sigma(b) = b\]
\[\tau(c) = cc\quad\sigma(c) = a\]

\[\tau(a) = abcc\]
\[\tau^2(a) = abccbccccc\]
\[\ldots\]
Morphic words : example

\[ \tau(a) = abcc \quad \quad \sigma(a) = a \]
\[ \tau(b) = b \quad \quad \quad \sigma(b) = b \]
\[ \tau(c) = cc \quad \quad \quad \sigma(c) = a \]

\[ \tau(a) = abcc \]
\[ \tau^2(a) = abcacbcccc \]
\[ \ldots \]
\[ \tau^\omega(a) = abcck \ldots c^2ib \ldots \]
Morphic words : example

\[ \tau(a) = abcc \quad \quad \quad \sigma(a) = a \]
\[ \tau(b) = b \quad \quad \quad \sigma(b) = b \]
\[ \tau(c) = cc \quad \quad \quad \sigma(c) = a \]

\[ \tau(a) = abcc \]
\[ \tau^2(a) = abccbcccc \]
\[ \ldots \]
\[ \tau^\omega(a) = abccb \ldots c^{2i}b \ldots \]
\[ \sigma(\tau^\omega(a)) = abaab \ldots a^{2i}b \ldots \]
First result

**Theorem**

$\omega$-frontiers of limit trees of (order-1) recursion schemes are exactly morphic words.
First result

Theorem
\[ \omega \text{-frontiers of limit trees of (order-1) recursion schemes are exactly morphic words.} \]
A nonterminal \( F \) has a \textbf{useless} parameter index \( i \) when \( x_i \) does not appear in \( [F \overrightarrow{x}] \).

Lemma (usefulness)
For any order-1 scheme, there is an order-1 scheme generating the same tree and where every nonterminal has only useful parameters.
First result

**Theorem**

\(\omega\)-frontiers of limit trees of (order-1) recursion schemes are exactly morphic words.

A nonterminal \(F\) has a **useless** parameter index \(i\) when \(x_i\) does not appear in \([F \xrightarrow{\rho}]\).

**Lemma (usefulness)**

For any order-1 scheme, there is an order-1 scheme generating the same tree and where every nonterminal has only useful parameters.

**Lemma (linearization)**

For any order-1 scheme \(S\) generating a comb, there is \(S'\) with only two nonterminals \(\{S, R\}\) such that \(\text{Fr}(S) = \text{Fr}(S')\). Moreover, each rewriting rule has exactly one occurrence of \(R\) and none of \(S\).
Proof sketch

\[ S \Rightarrow \]

\[ F \]

\[ F \Rightarrow \]

\[ x \quad y \]

\[ \sigma(\Delta) \quad \sigma(x) \quad \sigma(y) \quad \tau(x) \quad \tau(y) \]

\[ \{ c \in T \mid \rho(c) = 0 \} \cup \{ x, y, \Delta \} \]

\[ \tau(c) = c \text{ for all } c \in T \]

\[ \Delta \text{ is the root: } \tau(\Delta) = \Delta \]
Proof sketch

- Letters: \( \{ c \in T \mid \rho(c) = 0 \} \cup \{ x, y, \Delta \} \)
- \( \tau(c) = c \) for all \( c \in T \),
- \( \Delta \) is the root: \( \tau(\Delta) = \Delta u \)
Proof sketch

- Letters: \( \{ c \in T \mid \rho(c) = 0 \} \cup \{ x, y, \Delta \} \)
- \( \tau(c) = c \) for all \( c \in T \),
- \( \Delta \) is the root: \( \tau(\Delta) = \Delta u \)
Proof sketch

\[ S \implies F \quad \quad F \quad \implies x \]

Letters: \( \{ \Delta, a, b, x \} \)

\[
\begin{align*}
\sigma(\Delta) &= \varepsilon \\
\tau(\Delta) &= \Delta xb \\
\sigma(a) &= a \\
\tau(a) &= a \\
\sigma(b) &= b \\
\tau(b) &= b \\
\sigma(x) &= a \\
\tau(x) &= xx
\end{align*}
\]
Towards next-order morphic words

Can we expect to

- increase subword complexity?

**Theorem (Allouche-Shallit)**

*The number of words of length $n$ in a morphic word is at most $O(n^2)$.*

- increase growth rate?

**Theorem (Carton-Thomas)**

*The sequence of indexes of a given letter in a morphic word is at most $O(k^n)$ for some $k$.***
Recursion schemes: next order

Instead of simply arity, symbols have fixed type (starting with a base type \( o \)):

- **Terminals** \( T = \{ a, b, f, g \ldots \} \),
- **Nonterminals** \( N = \{ S, F, G \ldots \} \),
- A specific starting nonterminal \( S \)
- **Rewriting rules** for each nonterminal, using variables \( \mathcal{X} = \{ x, y, \phi, \psi \ldots \} \).
Recursion schemes: next order

Instead of simply arity, symbols have fixed type (starting with a base type $\text{o}$):

- terminals $T = \{a, b, f, g \ldots \}$,
- nonterminals $N = \{S, F, G \ldots \}$,
- a specific starting nonterminal $S$
- rewriting rules for each nonterminal, using variables $\mathcal{X} = \{x, y, \phi, \psi \ldots \}$.

\[
F \quad \Rightarrow \quad \phi \\
\phi \quad x \quad F \\
F \quad x \quad \phi \\
x : \text{o} \quad F : (\text{o} \rightarrow \text{o}) \rightarrow \text{o} \rightarrow \text{o} \\
\phi : \text{o} \rightarrow \text{o}
\]
Recursion schemes : next order

Instead of simply arity, symbols have fixed type (starting with a base type \( o \)):

- terminals \( T = \{a, b, f, g \ldots \} \),
- nonterminals \( N = \{S, F, G \ldots \} \),
- a specific starting nonterminal \( S \)
- rewriting rules for each nonterminal, using variables \( \mathcal{X} = \{x, y, \phi, \psi \ldots \} \).

\[
\begin{align*}
F & \Rightarrow \phi \\
\phi & \ x \\
\phi & \ F \ x \\
F & \ x \\
\phi & \\
\end{align*}
\]

\[
\begin{align*}
x & : o \\
F & : (o \to o) \to o \to o \\
\phi & : o \to o \\
F\phi & : o \to o
\end{align*}
\]
The Champernowne’s constant is simply the concatenation of numbers.

012345677891011...  
011011100101110...
Champernowne : scheme approach

\[
S \quad \Rightarrow \quad f
\]

\[
G \quad \Rightarrow \quad f
\]

\[
F \quad \Rightarrow \quad f
\]

\[
S : \quad o \\
F : \quad (o \rightarrow o) \rightarrow o \\
G : \quad (o \rightarrow o) \rightarrow o \rightarrow o
\]

\[
f : \quad o \rightarrow o \rightarrow o \\
0 : \quad o \\
1 : \quad o
\]
Champernowne : scheme approach

\[ S \Rightarrow f \]

\[ G \Rightarrow f \]

\[ F \Rightarrow f \]

\[ S \Rightarrow^2 f \]

\[ \phi \]

\[ x \]

\[ 0 \]

\[ f \]

\[ 1 \]

\[ F \]

\[ \phi \]

\[ x \]

\[ 0 \]

\[ x \]

\[ 1 \]

\[ \phi \]

\[ F \]

\[ \phi \]

\[ \phi \]

\[ \phi \]

\[ G \]

\[ 0 \]

\[ 1 \]

\[ \phi \]

\[ f \]

\[ 1 \]
Champernowne: scheme approach
Champernowne : scheme approach
Linearization of order-2 schemes

Can we have the same linearization lemma as before?
Linearization of order-2 schemes

Can we have the same linearization lemma as before?

**Lemma**

For any scheme in $S_2$, there is a scheme in $S_2$ with only useful nonterminals generating the same tree.

$$
\begin{array}{c}
\phi \\
\downarrow \\
H
\end{array} \Rightarrow 
\begin{array}{c}
\phi \\
\downarrow \\
F \\
\downarrow \\
G \\
\downarrow \\
x
\end{array}
\begin{array}{c}
x \\
\downarrow \\
H
\end{array}
\begin{array}{c}
H \\
\phi \\
x
\end{array} 
$$

($H \rightarrow a_i$ means “argument $i$ has arity reduced by $a_i$”)
Linearization of order-2 schemes

Can we have the same linearization lemma as before?

Lemma
For any scheme in $S_2$, there is a scheme in $S_2$ with only useful nonterminals generating the same tree.
Linearization of order-2 schemes

Can we have the same linearization lemma as before?

Lemma

For any scheme in $S_2$, there is a scheme in $S_2$ with only useful nonterminals generating the same tree.

$(H \xrightarrow{a} \text{ means “argument } i \text{ has arity reduced by } a_i)$
Nonterminals are separated into

- *semiterminals*: nonterminals rewriting into finite trees,
- *∞-nonterminals*: the other ones.

**Lemma (linearization)**

For any order-2 scheme $\mathcal{S}$ generating a comb, there is $\mathcal{S}'$ with only two $∞$-nonterminals $\{S, R\}$ such that $\text{Fr}(\mathcal{S}) = \text{Fr}(\mathcal{S}')$. Moreover, their rewriting rules have exactly one occurrence of $R$ and none of $S$. 
Linearization

Given the shape of the tree, we actually never have two $\infty$-nonterminals at the same time.
Linearization

Given the shape of the tree, we actually never have two $\infty$-nonterminals at the same time.

\[
S \Rightarrow^* \begin{array}{c}
\vdots \\
F \\
G \\
t^1_G
\end{array} \Rightarrow^* \begin{array}{c}
\vdots \\
G \\
F' \\
t^1_G
\end{array} \text{ or } \begin{array}{c}
\vdots \\
G \\
F'
\end{array}
\]
Linearization

Given the shape of the tree, we actually never have two $\infty$-nonterminals at the same time.
Term words

Alphabet $\Sigma = \bigcup_{i=0}^{n} \Sigma_i$ where $\Sigma_0$ is called "letters".

$$\theta := \epsilon \mid a \in \Sigma_0 \mid f(\theta, \ldots, \theta), f \in \Sigma_i \mid \theta \cdot \theta_{i}$$
Term words

Alphabet $\Sigma = \bigcup_{i=0}^{n} \Sigma_i$ where $\Sigma_0$ is called "letters".

$$\theta := \epsilon \mid a \in \Sigma_0 \mid f(\underbrace{\theta, \ldots, \theta}_i), f \in \Sigma_i \mid \theta \cdot \theta$$

We use variables from $\mathcal{V} = \{z_1, \ldots\}$ to define $\Sigma(\bar{z}) = \{f(z_1, \ldots, z_k) \mid f \in \Sigma_k\}$. 
Term words

Alphabet $\Sigma = \bigcup_{i=0}^{n} \Sigma_i$ where $\Sigma_0$ is called "letters".

$$\theta := \epsilon \mid a \in \Sigma_0 \mid f(\theta, \ldots, \theta), f \in \Sigma_i \mid \theta \cdot \theta$$

We use variables from $\mathcal{V} = \{z_1, \ldots\}$ to define $\Sigma(\bar{z}) = \{f(z_1, \ldots, z_k) \mid f \in \Sigma_k\}$.

Let $\tau, \sigma$ be two morphisms on $\Sigma(\bar{z})^*$ w.r.t. concatenation.

for $f \in \Sigma_k$ and $z_1, \ldots, z_k \in \mathcal{V}$,

$$\tau(f(z_1, \ldots, z_k)) \in \text{TW}(\Sigma \cup \{z_1, \ldots, z_k\})$$

$$\sigma(f(z_1, \ldots, z_k)) \in \text{TW}(\Sigma_0 \cup \{z_1, \ldots, z_k\})$$

$$= (\Sigma_0 \cup \{z_1, \ldots, z_k\})^*$$
This definition is extended on term words by

\[
\text{for } f \in \Sigma_k \\
\text{and } t_1, \ldots, t_k \in TW(\Sigma), \\
\tau(f(t_1, \ldots, t_k)) = \tau(f(z_1, \ldots, z_k)) [\forall i, z_i := \tau(t_i)] \\
\sigma(f(t_1, \ldots, t_k)) = \sigma(f(z_1, \ldots, z_k)) [\forall i, z_i := \sigma(t_i)]
\]

Let \( \Delta \in \Sigma_0 \), words of the form \( \sigma(\tau^\omega(\Delta)) \) are \textit{2-morphic words}. 
\( \Sigma_0 = \{0, 1\} \) and \( \Sigma_1 = \{g\} \).

\[
\begin{align*}
\tau(\Delta) &= \Delta g(0) g(1) \\
\tau(g(z)) &= g(z0) g(z1) \\
\sigma(\Delta) &= 01 \\
\sigma(g(z)) &= 1z
\end{align*}
\]

In addition \( \tau(1) = \sigma(1) = 1 \) and \( \tau(0) = \sigma(0) = 0 \).

\[
\tau(\Delta) = \Delta \ g(0) \ g(1)
\]
Champernowne : 2-morphic words

\[ \Sigma_0 = \{0, 1\} \text{ and } \Sigma_1 = \{g\}. \]

\[
\begin{align*}
\tau(\Delta) & = \Delta g(0)g(1) \\
\tau(g(z)) & = g(z0)g(z1) \\
\sigma(\Delta) & = 01 \\
\sigma(g(z)) & = 1z
\end{align*}
\]

In addition \( \tau(1) = \sigma(1) = 1 \) and \( \tau(0) = \sigma(0) = 0. \)

\[
\begin{align*}
\tau(\Delta) & = \Delta g(0) g(1) \\
\tau^{(2)}(\Delta) & = \Delta g(0) g(1) g(00) g(01) g(10) g(11)
\end{align*}
\]
Champernowne: 2-morphic words

\[ \Sigma_0 = \{0, 1\} \text{ and } \Sigma_1 = \{g\}. \]

\[
\begin{align*}
\tau(\Delta) &= \Delta g(0) g(1) \\
\tau(g(z)) &= g(z0) g(z1) \\
\sigma(\Delta) &= 01 \\
\sigma(g(z)) &= 1z
\end{align*}
\]

In addition \( \tau(1) = \sigma(1) = 1 \) and \( \tau(0) = \sigma(0) = 0. \)

\[
\begin{align*}
\tau(\Delta) &= \Delta \ g(0) \ g(1) \\
\tau^{(2)}(\Delta) &= \Delta \ g(0) \ g(1) \ g(00) \ g(01) \ g(10) \ g(11) \\
\sigma(\tau^{(2)}(\Delta)) &= 01 \ 10 \ 11 \ 100 \ 101 \ 110 \ 111
\end{align*}
\]
Final result

Theorem
The frontiers of combs generated by order-2 schemes are exactly 2-morphic words.
Theorem

The frontiers of combs generated by order-2 schemes are exactly 2-morphic words.

A safe scheme: in every rule $F \xrightarrow{x} T_F$, and every subterm $t$ of $T_F$, the order of $t$ is lower of equal to any order of $x_i$ inside it. The proof of the theorem translates words into safe schemes.
Final result

Theorem

The frontiers of combs generated by order-2 schemes are exactly 2-morphic words.

A safe scheme: in every rule $F \overrightarrow{x} \Rightarrow T_F$, and every subterm $t$ of $T_F$, the order of $t$ is lower or equal to any order of $x_i$ inside it. The proof of the theorem translates words into safe schemes.

Moreover, by MSO properties of the pushdown hierarchy, 2-morphic words are also

- $\omega$-frontiers of safe trees,
- paths generated by order-3 safe schemes.

What about unsafe ones?
Consequences

We apply properties of the pushdown hierarchy.

Corollary

For any 2-morphic word $w$,

- the MSO theory of $w$ is decidable;
- for any MSO-transduction $\mathcal{T}$, if $\mathcal{T}(w)$ is a word, it is a 2-morphic word.
- the sequence of indexes of a given letter in a morphic word is at most $\mathcal{O}(2^{2^Cn})$ for some $C$. The bound is tight.

Other example: characteristic word of $(n!)_{n \geq 0}$, known as the Liouville constant.
Conclusion and beyond

This construction builds a new class of graphs for order-2 schemes.

▶ What about higher orders? can we still linearize?
▶ Connexion with classes $S_k$ of [Fratani-Senizergues], or $k$-automatic words by [Bárány]?
▶ And beyond the pushdown/Caucal/scheme hierarchy?

the characteristic word of

$$\left(2^{2^{\ddots}}\right)_n$$

has decidable MSO-theory [Thomas]