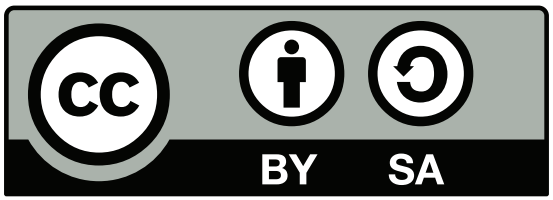
Parallel Hyperedge Replacement String Languages

11th International Workshop on Computing with Terms and Graphs

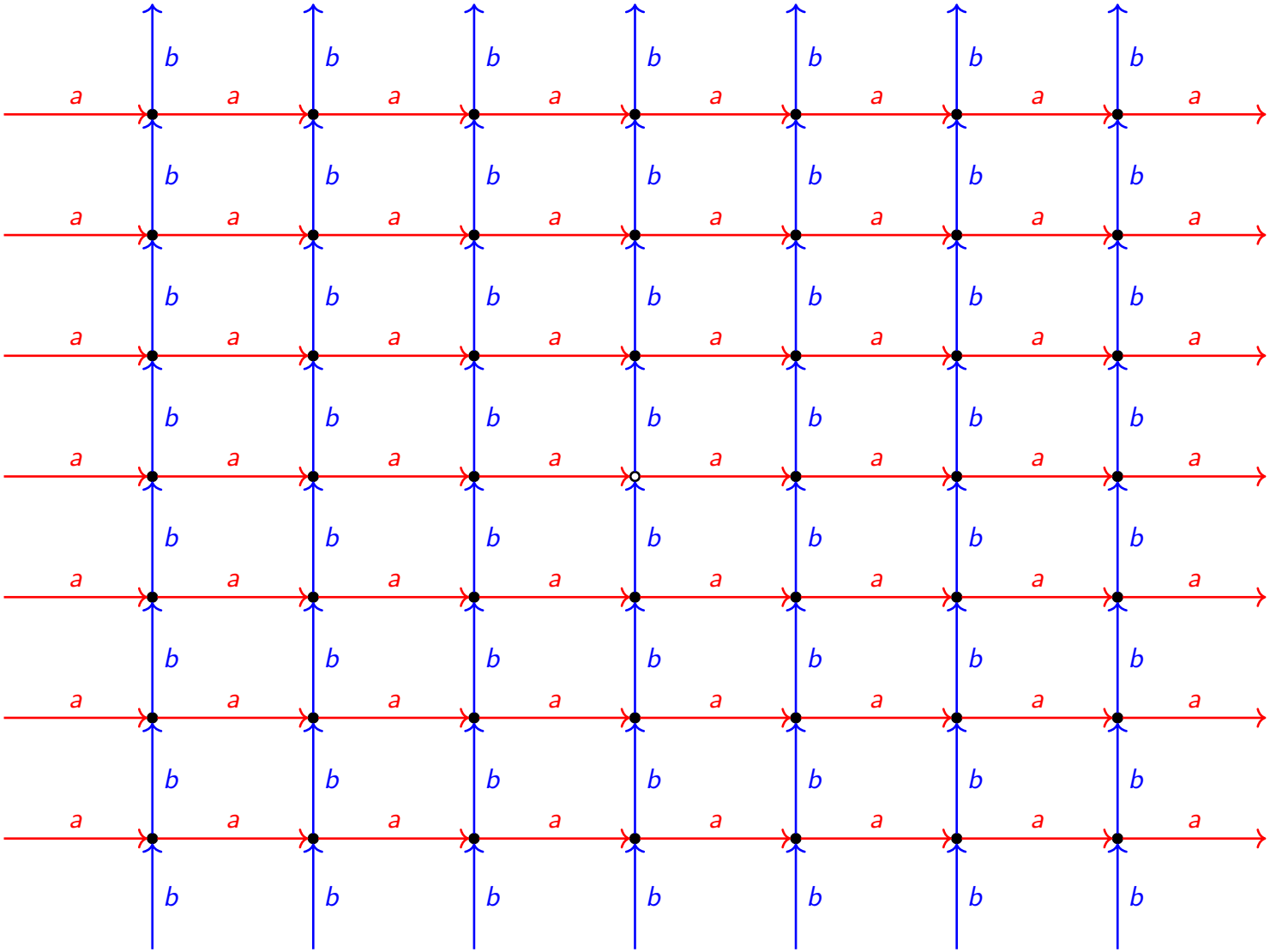
Graham Campbell

School of Mathematics, Statistics and Physics, Newcastle University, UK

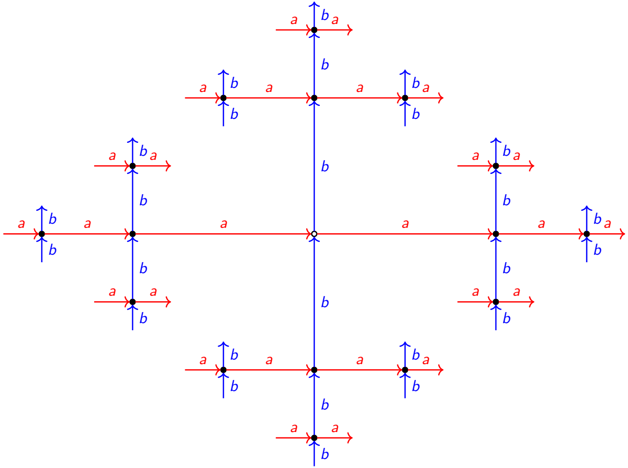
July 2020



Cayley Graph of $\mathbb{Z}^2$



Cayley Graph of F_2



The Word Problem

Definition (Word Problem)

Given a group presentation $\langle X \mid R \rangle$ for a group G, the word problem is the membership problem for the string language:

$$\mathrm{WP}_X(G) = \{w \in (X \cup X^{-1})^* \mid w =_G 1_G\}.$$

The Word Problem

Definition (Word Problem)

Given a group presentation $\langle X \mid R \rangle$ for a group G, the word problem is the membership problem for the string language:

$$\mathrm{WP}_X(G) = \{w \in (X \cup X^{-1})^* \mid w =_G 1_G\}.$$

Question: How hard is the word problem?

The Word Problem

Definition (Word Problem)

Given a group presentation $\langle X \mid R \rangle$ for a group G, the word problem is the membership problem for the string language:

$$\mathrm{WP}_X(G) = \{w \in (X \cup X^{-1})^* \mid w =_G 1_G\}.$$

Question: How hard is the word problem?

Answer: It is impossible! There is a finite group presentation with an undecidable word problem (Novikov 1955).

The Word Problem

Definition (Word Problem)

Given a group presentation $\langle X \mid R \rangle$ for a group G, the word problem is the membership problem for the string language:

$$\mathrm{WP}_X(G) = \{w \in (X \cup X^{-1})^* \mid w =_G 1_G\}.$$

Question: How hard is the word problem?

Answer: It is impossible! There is a finite group presentation with an undecidable word problem (Novikov 1955).

Question: The Chomsky hierarchy is a notion of complexity of a language. Can we characterise the finitely generated groups in terms of language families?

The Word Problem

Definition (Word Problem)

Given a group presentation $\langle X \mid R \rangle$ for a group G, the word problem is the membership problem for the string language:

$$\mathrm{WP}_X(G) = \{w \in (X \cup X^{-1})^* \mid w =_G 1_G\}.$$

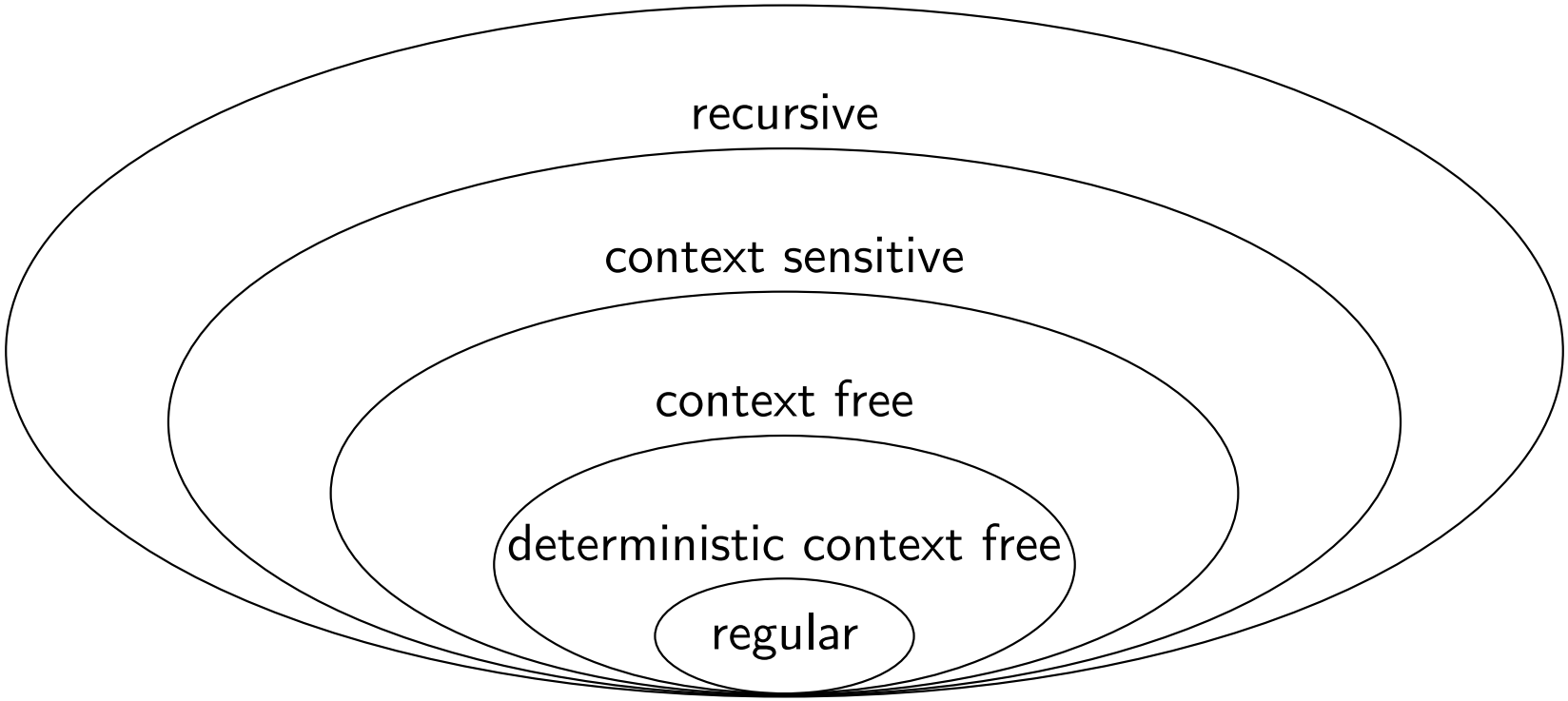
Question: How hard is the word problem?

Answer: It is impossible! There is a finite group presentation with an undecidable word problem (Novikov 1955).

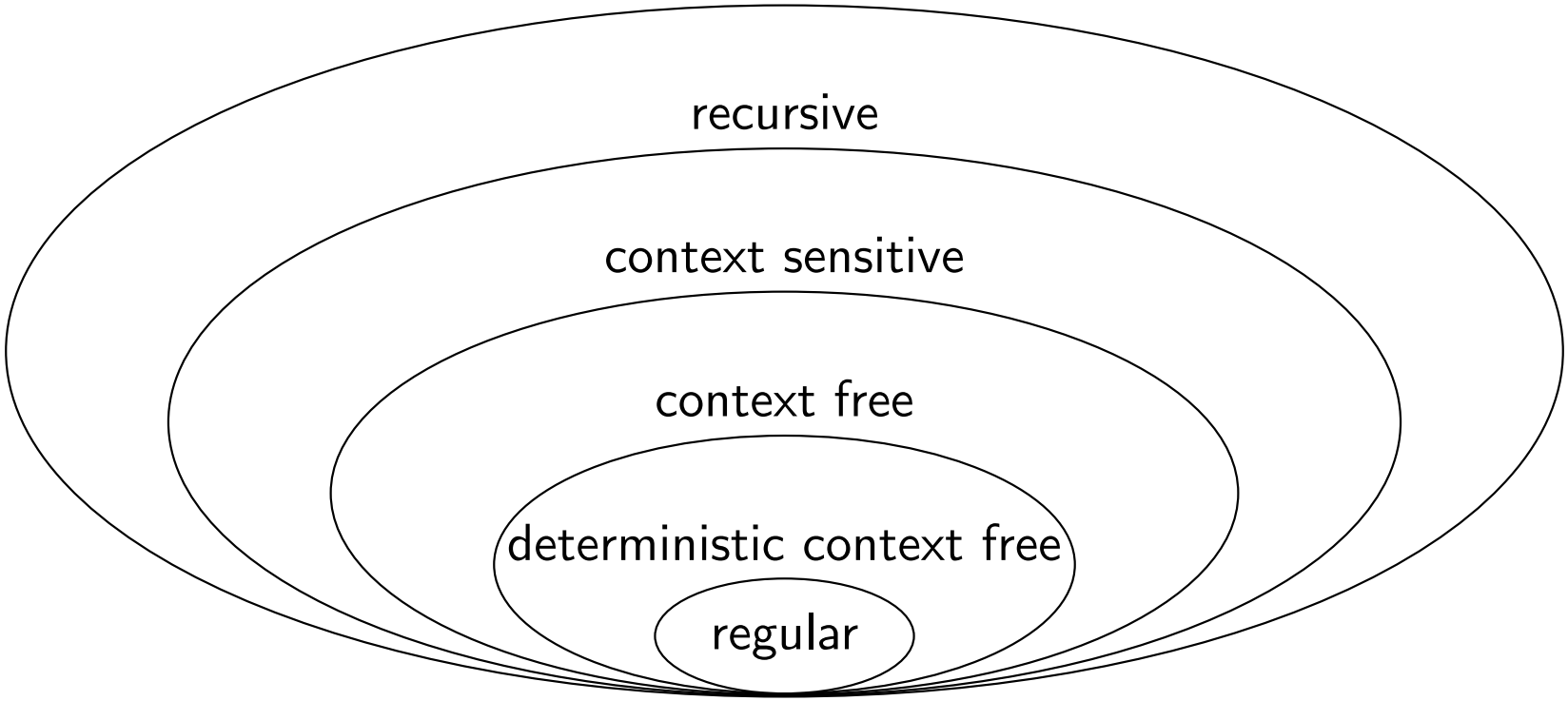
Question: The Chomsky hierarchy is a notion of complexity of a language. Can we characterise the finitely generated groups in terms of language families?

Answer: Maybe...

Context-Free Groups



Context-Free Groups



Theorem (Anisimov (1971) and Muller and Schupp (1983))

1. A presentation defines a finite group iff it has regular WP;
2. A presentation defines a virtually free group iff it has (D)CF WP.

Multiple Context-Free

Question: how "far away" are interesting families of groups from the context-free languages in the formal language formal language hierarchy?

Multiple Context-Free

Question: how "far away" are interesting families of groups from the context-free languages in the formal language formal language hierarchy?

MCF languages are a conservative extension of the context-free languages, formalised by Seki, Matsumura, Fujii, and Kasami 1991.

Multiple Context-Free

Question: how "far away" are interesting families of groups from the context-free languages in the formal language formal language hierarchy?

MCF languages are a conservative extension of the context-free languages, formalised by Seki, Matsumura, Fujii, and Kasami 1991.

Theorem (Ho (2018))

If a presentation defines a virtually Abelian group, then it has MCF WP.

Multiple Context-Free

Question: how "far away" are interesting families of groups from the context-free languages in the formal language formal language hierarchy?

MCF languages are a conservative extension of the context-free languages, formalised by Seki, Matsumura, Fujii, and Kasami 1991.

Theorem (Ho (2018))

If a presentation defines a virtually Abelian group, then it has MCF WP.

Theorem (Gilman, Kropholler, and Schleimer (2018))

The fundamental group of a hyperbolic three-manifold does not admit a MCF WP.

Multiple Context-Free

Question: how "far away" are interesting families of groups from the context-free languages in the formal language formal language hierarchy?

MCF languages are a conservative extension of the context-free languages, formalised by Seki, Matsumura, Fujii, and Kasami 1991.

Theorem (Ho (2018))

If a presentation defines a virtually Abelian group, then it has MCF WP.

Theorem (Gilman, Kropholler, and Schleimer (2018))

The fundamental group of a hyperbolic three-manifold does not admit a MCF WP.

Theorem (Engelfriet and Heyker (1991) and Weir (1992))

The MCF languages are exactly the string languages generated by HR grammars (Drewes, Kreowski, and Habel 1997).

Lindenmayer Systems

There lots of other well-behaved language classes sitting in between the CF and CS classes, such as the indexed languages (Aho 1968) and their subclass of ET0L languages (Rozenberg and Salomaa 1980).

Lindenmayer Systems

There lots of other well-behaved language classes sitting in between the CF and CS classes, such as the indexed languages (Aho 1968) and their subclass of ET0L languages (Rozenberg and Salomaa 1980).

It is not known if there are any groups with indexed word problems other than the virtually free groups.

Lindenmayer Systems

There lots of other well-behaved language classes sitting in between the CF and CS classes, such as the indexed languages (Aho 1968) and their subclass of ET0L languages (Rozenberg and Salomaa 1980).

It is not known if there are any groups with indexed word problems other than the virtually free groups.

In particular, we don't know if any hyperbolic groups (other than the virtually free groups) have ET0L word problems (Ciobanu, Elder, and Ferov 2018), such as the fundamental group of the double torus.

Lindenmayer Systems

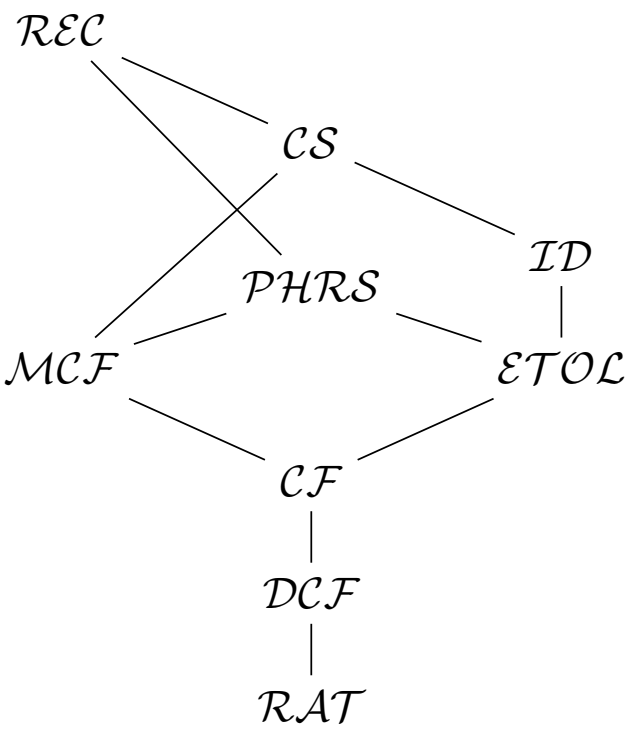
There lots of other well-behaved language classes sitting in between the CF and CS classes, such as the indexed languages (Aho 1968) and their subclass of ET0L languages (Rozenberg and Salomaa 1980).

It is not known if there are any groups with indexed word problems other than the virtually free groups.

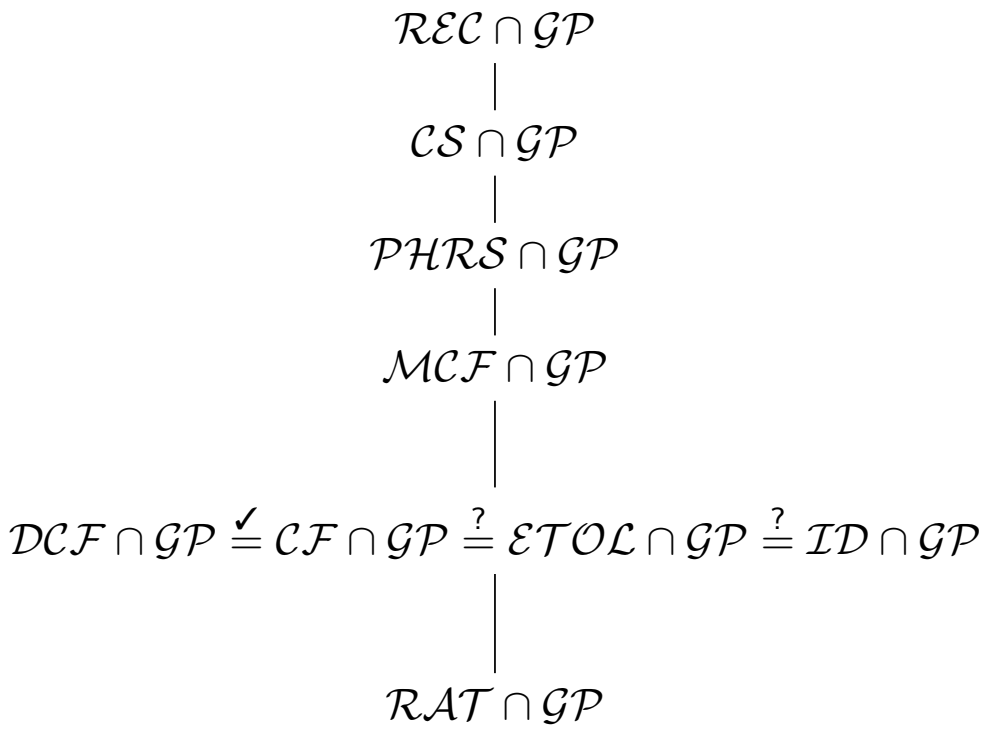
In particular, we don't know if any hyperbolic groups (other than the virtually free groups) have ET0L word problems (Ciobanu, Elder, and Ferov 2018), such as the fundamental group of the double torus.

What if we tried to mix together ideas from MCF and ET0L. . . parallel hyperedge replacement! Do the word problems of hyperbolic groups lie within this class? I should mention forms of parallel HR have been considered before (Habel 1992; Kreowski 1993), the work is not extensive and does not consider rational control or string generational power.

Taking Stock

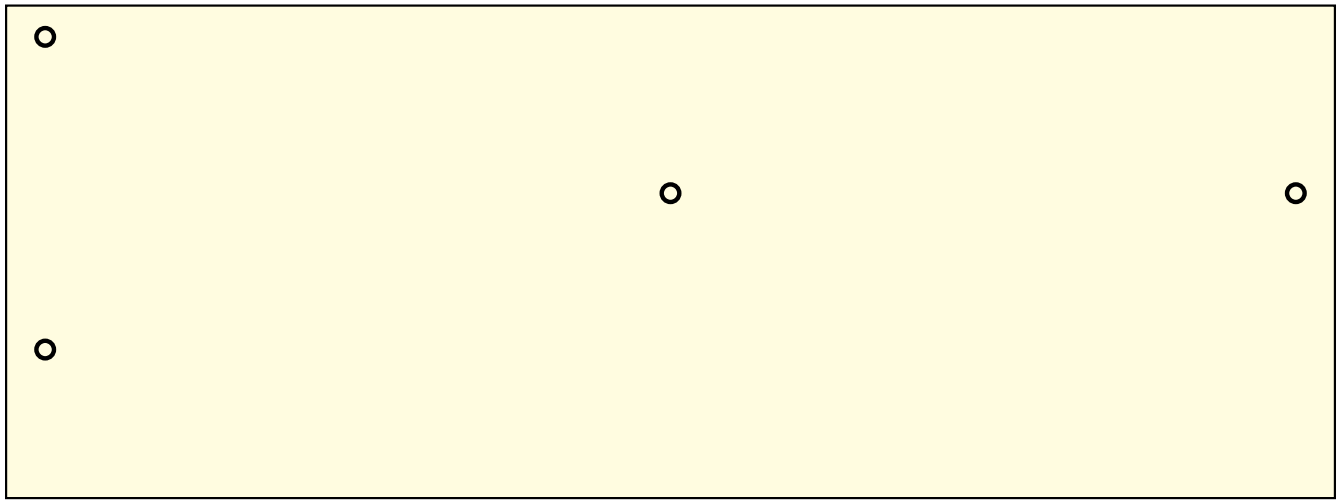


(a) Proved String Language Hierarchy

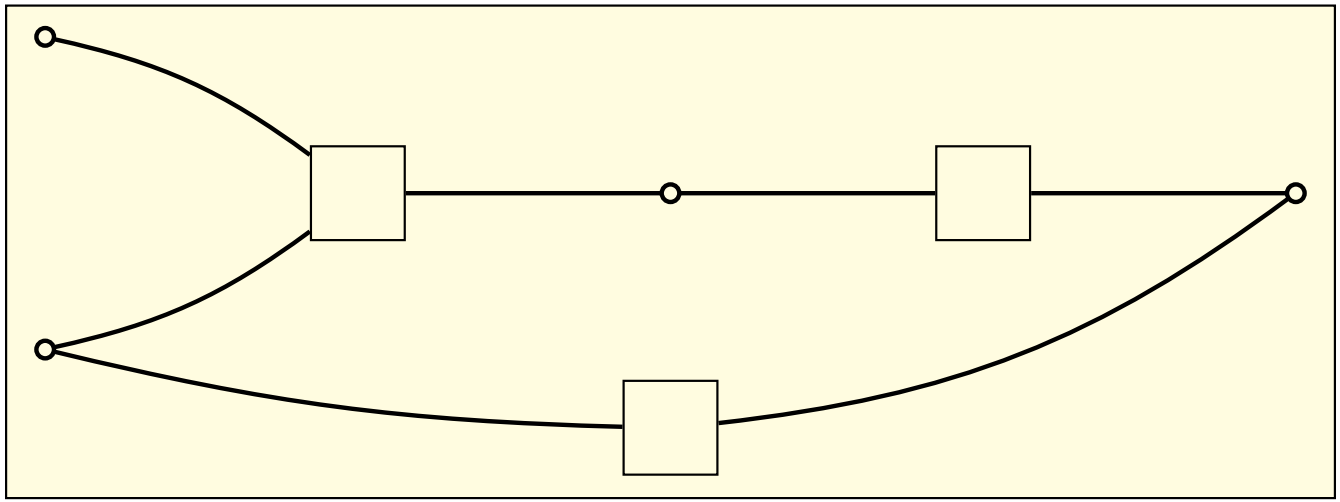


(b) Conjectured Group Language Hierarchy

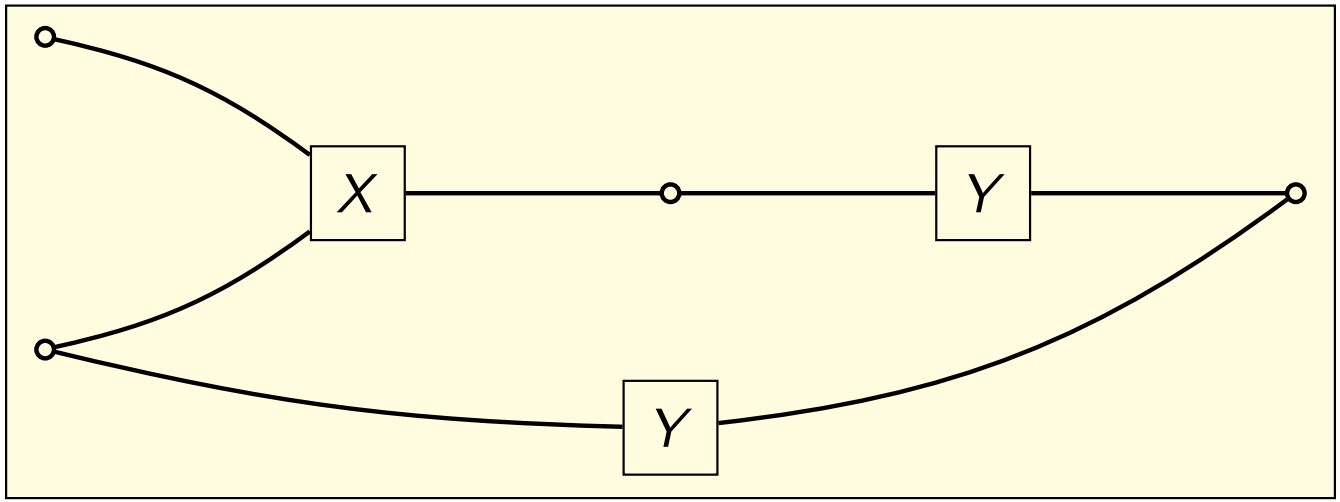
Hypergraphs I



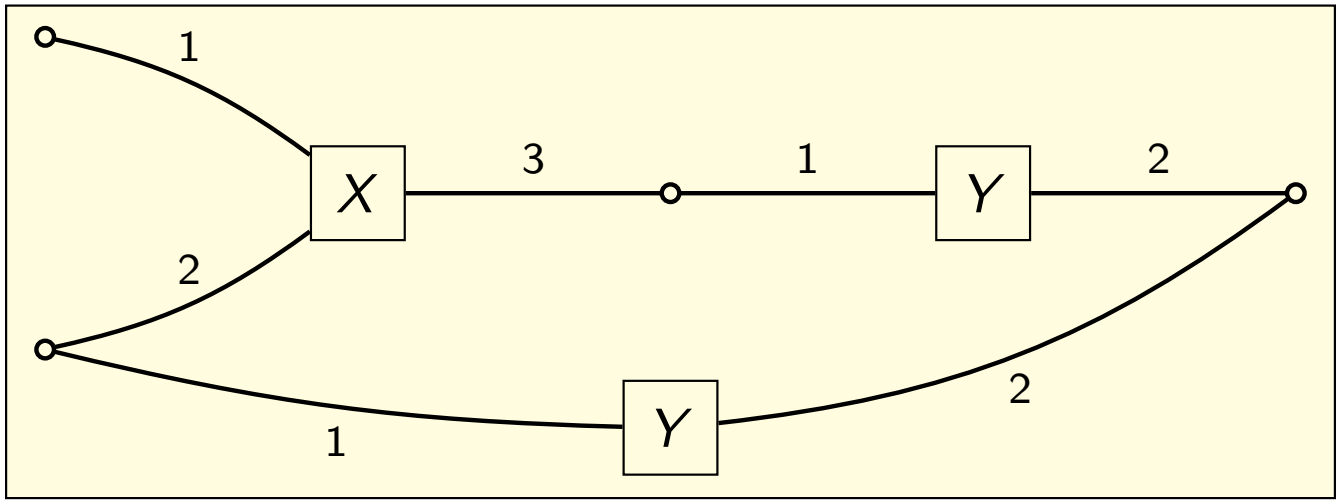
Hypergraphs I



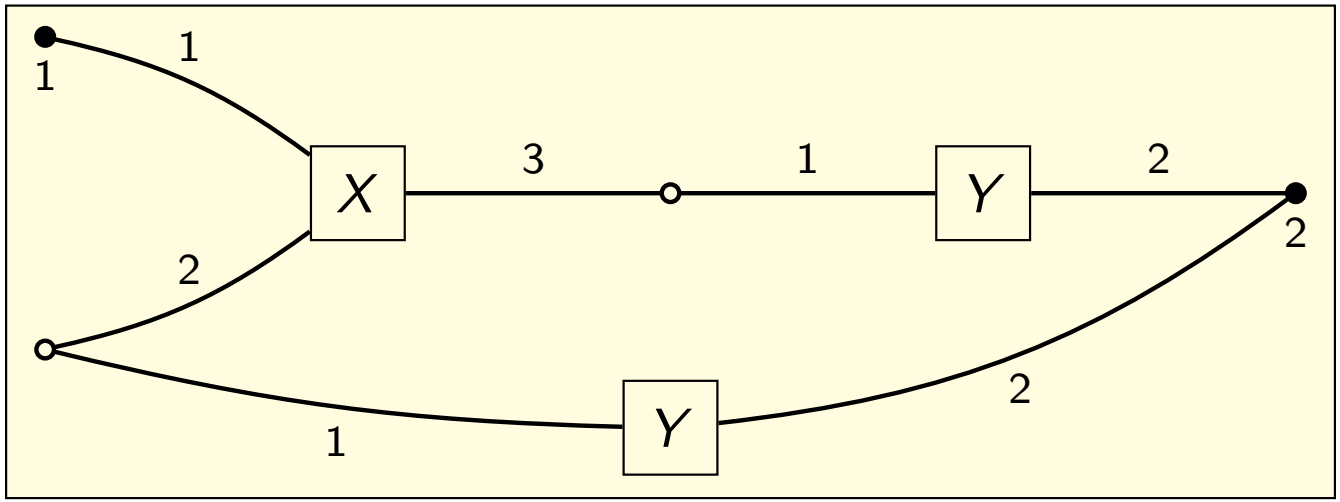
Hypergraphs I



Hypergraphs I



Hypergraphs I



Hypergraphs II

Definition (Signature)

A signature is a pair $\mathcal{C} = (\Sigma, \mathrm{type})$ where Σ is some finite label set, and $\mathrm{type} : \Sigma \to \mathbb{N}$ is a typing function which assigns to each label an arity.

Hypergraphs II

Definition (Signature)

A signature is a pair $\mathcal{C} = (\Sigma, \mathrm{type})$ where Σ is some finite label set, and $\mathrm{type} : \Sigma \to \mathbb{N}$ is a typing function which assigns to each label an arity.

Definition (Hypergraph)

A hypergraph over $\mathcal{C}$ is a tuple $H = (V_H, E_H, \mathrm{att}_H, \mathrm{lab}_H, \mathrm{ext}_H)$ where:

1. V_H is a finite set of nodes;
2. E_H is a finite set of hyperedges;
3. $\mathrm{att}_H : E_H \to \mathrm{seq}(V_H)$ is the attachment function;
4. $\mathrm{lab}_H : E_H \to \Sigma$ is the labelling function;
5. $\mathrm{ext}_H : \mathrm{iseq}(V_H)$ are the external nodes;

such that labelling is compatible with typing ($\mathrm{type} \circ \mathrm{lab}_H = |\cdot| \circ \mathrm{att}_H$).

Hypergraphs II

Definition (Signature)

A signature is a pair $\mathcal{C} = (\Sigma, \mathrm{type})$ where Σ is some finite label set, and $\mathrm{type} : \Sigma \to \mathbb{N}$ is a typing function which assigns to each label an arity.

Definition (Hypergraph)

A hypergraph over $\mathcal{C}$ is a tuple $H = (V_H, E_H, \mathrm{att}_H, \mathrm{lab}_H, \mathrm{ext}_H)$ where:

1. V_H is a finite set of nodes;
2. E_H is a finite set of hyperedges;
3. $\mathrm{att}_H : E_H \to \mathrm{seq}(V_H)$ is the attachment function;
4. $\mathrm{lab}_H : E_H \to \Sigma$ is the labelling function;
5. $\mathrm{ext}_H : \mathrm{iseq}(V_H)$ are the external nodes;

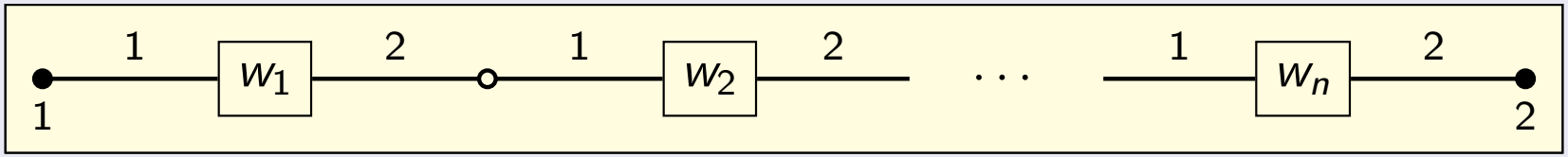
such that labelling is compatible with typing ($\mathrm{type} \circ \mathrm{lab}_H = |\cdot| \circ \mathrm{att}_H$).

The class of all hypergraphs over $\mathcal{C}$ is denoted $\mathcal{H}_{\mathcal{C}}$.

Strings and Handles

Definition (String Graph)

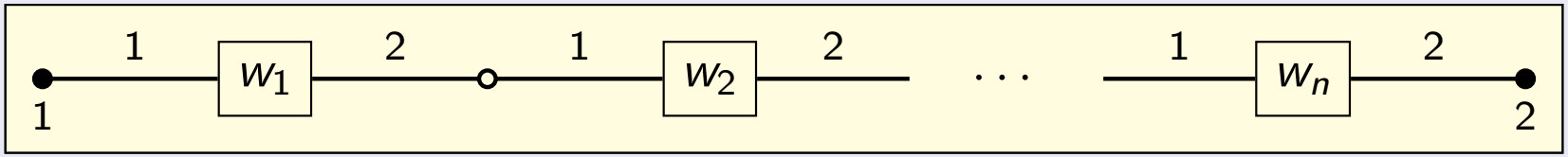
Given a non-empty word $w = w_1 w_2 \cdots w_n$, we define its string graph $w^\bullet$:



Strings and Handles

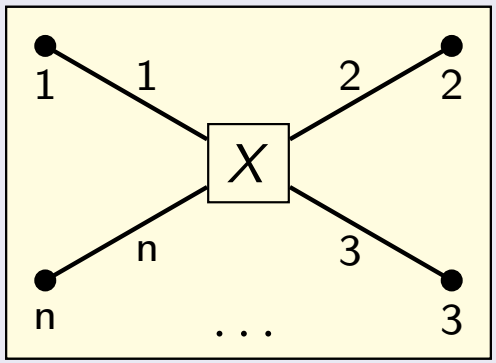
Definition (String Graph)

Given a non-empty word $w = w_1 w_2 \cdots w_n$, we define its string graph $w^\bullet$:

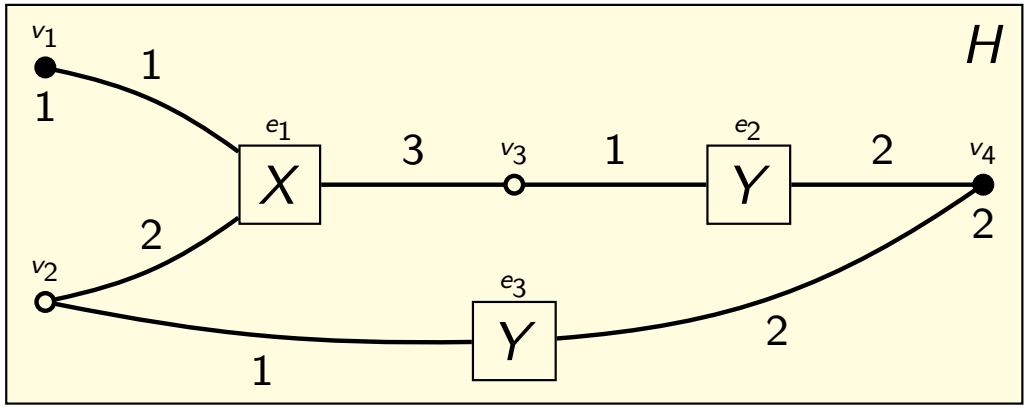


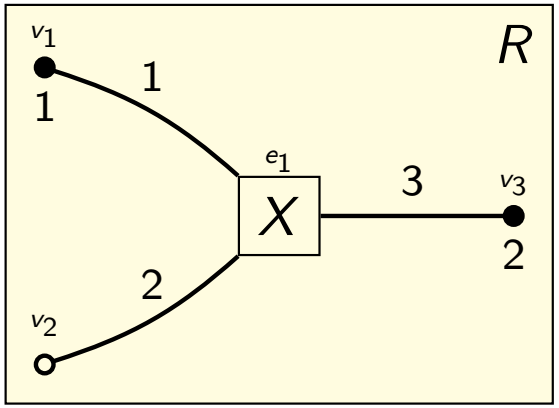
Definition (Handle)

Given a label X of type n, we define its handle $X^\bullet$:

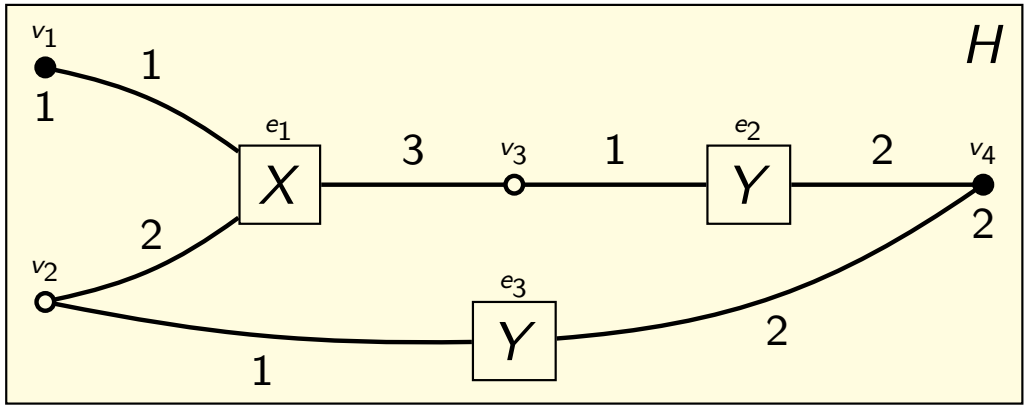


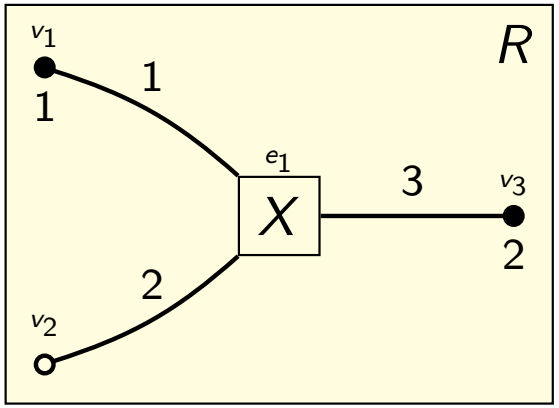
Replacement

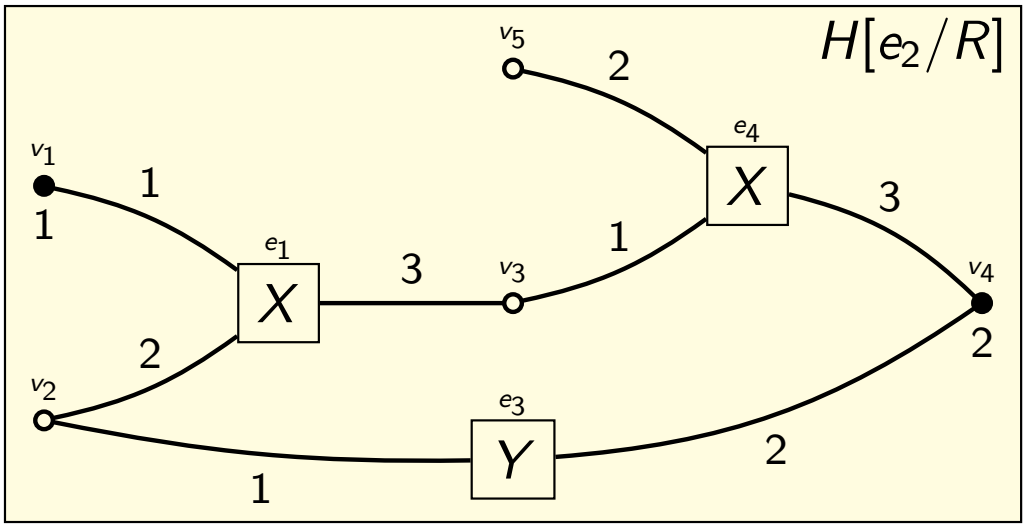




Replacement







Rules and Derivations

Let $\mathcal{C} = (\Sigma, \mathrm{type})$ be a signature and $N \subseteq \Sigma$ a set of non-terminals.

Rules and Derivations

Let $\mathcal{C} = (\Sigma, \mathrm{type})$ be a signature and $N \subseteq \Sigma$ a set of non-terminals.

Definition (Rule)

A rule over N is a pair (L, R) with $L \in N$, $R \in \mathcal{H}_{\mathcal{C}}$, $\mathrm{type}(L) = \mathrm{type}(R)$.

Rules and Derivations

Let $\mathcal{C} = (\Sigma, \mathrm{type})$ be a signature and $N \subseteq \Sigma$ a set of non-terminals.

Definition (Rule)

A rule over N is a pair (L, R) with $L \in N$, $R \in \mathcal{H}_{\mathcal{C}}$, $\mathrm{type}(L) = \mathrm{type}(R)$.

Definition (Direct Derivation)

Given $H \in \mathcal{H}_{\mathcal{C}}$ and $\mathcal{R}$ a set of rules, if $e \in E_H$ and $(\mathrm{lab}_H(e), R) \in \mathcal{R}$, then we say that H directly derives $H' \cong H[e/R]$, and write $H \Rightarrow_{\mathcal{R}} H'$.

Rules and Derivations

Let $\mathcal{C} = (\Sigma, \mathrm{type})$ be a signature and $N \subseteq \Sigma$ a set of non-terminals.

Definition (Rule)

A rule over N is a pair (L, R) with $L \in N$, $R \in \mathcal{H}_{\mathcal{C}}$, $\mathrm{type}(L) = \mathrm{type}(R)$.

Definition (Direct Derivation)

Given $H \in \mathcal{H}_{\mathcal{C}}$ and $\mathcal{R}$ a set of rules, if $e \in E_H$ and $(\mathrm{lab}_H(e), R) \in \mathcal{R}$, then we say that H directly derives $H' \cong H[e/R]$, and write $H \Rightarrow_{\mathcal{R}} H'$.

Definition (Derivation)

H derives H' if there is a sequence $H \Rightarrow_{\mathcal{R}} H_1 \Rightarrow_{\mathcal{R}} \cdots \Rightarrow_{\mathcal{R}} H_k \cong H'$ for some $k \in \mathbb{N}$. We write $H \Rightarrow^k_{\mathcal{R}} H'$ or $H \Rightarrow^*_{\mathcal{R}} H'$.

HR Grammars

Definition (HR Grammar)

A HR grammar of order k is a system $\mathcal{G} = (\mathcal{C}, N, S, \mathcal{R})$ where:

1. $\mathcal{C} = (\Sigma, \mathrm{type})$ is a signature;
2. $N \subseteq \Sigma$ is the set of non-terminal labels;
3. $S \in N$ is the start symbol;
4. $\mathcal{R}$ is a finite set of rules over N;

with $\max(\{\mathrm{type}(R) \mid (L, R) \in \mathcal{R}\}) \leq k$. The generated language is:

$$\mathrm{L}(\mathcal{G}) = \{H \in \mathcal{H}_{\mathcal{C}} \mid S^{\bullet} \Rightarrow^{*}_{\mathcal{R}} H \text{ with } \mathrm{lab}_{H}^{-1}(N) = \emptyset\} \subseteq \mathcal{H}_{\mathcal{C}}.$$

HR Grammars

Definition (HR Grammar)

A HR grammar of order k is a system $\mathcal{G} = (\mathcal{C}, N, S, \mathcal{R})$ where:

1. $\mathcal{C} = (\Sigma, \mathrm{type})$ is a signature;
2. $N \subseteq \Sigma$ is the set of non-terminal labels;
3. $S \in N$ is the start symbol;
4. $\mathcal{R}$ is a finite set of rules over N;

with $\max(\{\mathrm{type}(R) \mid (L, R) \in \mathcal{R}\}) \leq k$. The generated language is:

$$\mathrm{L}(\mathcal{G}) = \{H \in \mathcal{H}_{\mathcal{C}} \mid S^{\bullet} \Rightarrow^{*}_{\mathcal{R}} H \text{ with } \mathrm{lab}_{H}^{-1}(N) = \emptyset\} \subseteq \mathcal{H}_{\mathcal{C}}.$$

$L \subseteq \mathcal{H}_{\mathcal{C}}$ is called a HR language of order k (k-HR language) if there is a k-HR grammar such that $\mathrm{L}(\mathcal{G}) = L$.

HR Grammars

Definition (HR Grammar)

A HR grammar of order k is a system $\mathcal{G} = (\mathcal{C}, N, S, \mathcal{R})$ where:

1. $\mathcal{C} = (\Sigma, \mathrm{type})$ is a signature;
2. $N \subseteq \Sigma$ is the set of non-terminal labels;
3. $S \in N$ is the start symbol;
4. $\mathcal{R}$ is a finite set of rules over N;

with $\max(\{\mathrm{type}(R) \mid (L, R) \in \mathcal{R}\}) \leq k$. The generated language is:

$$\mathrm{L}(\mathcal{G}) = \{H \in \mathcal{H}_{\mathcal{C}} \mid S^{\bullet} \Rightarrow^{*}_{\mathcal{R}} H \text{ with } \mathrm{lab}_{H}^{-1}(N) = \emptyset\} \subseteq \mathcal{H}_{\mathcal{C}}.$$

$L \subseteq \mathcal{H}_{\mathcal{C}}$ is called a HR language of order k (k-HR language) if there is a k-HR grammar such that $\mathrm{L}(\mathcal{G}) = L$.

The class of HR languages is the union of all k-HR languages for $k \in \mathbb{N}$.

HRS Languages

Definition (String Graph Language)

A hypergraph language that only consists of string graphs is called a string graph language. We can identify the language members with the strings they represent.

HRS Languages

Definition (String Graph Language)

A hypergraph language that only consists of string graphs is called a string graph language. We can identify the language members with the strings they represent.

Thus, a string language L is called a HRS language if, up to treatment of the empty string, there is a HR grammar that generates the language of string graphs that represent exactly the strings in L.

HRS Languages

Definition (String Graph Language)

A hypergraph language that only consists of string graphs is called a string graph language. We can identify the language members with the strings they represent.

Thus, a string language L is called a HRS language if, up to treatment of the empty string, there is a HR grammar that generates the language of string graphs that represent exactly the strings in L.

The class of HRS languages is the union of all k-HRS languages for $k \in \mathbb{N}$.

HRS Languages

Definition (String Graph Language)

A hypergraph language that only consists of string graphs is called a string graph language. We can identify the language members with the strings they represent.

Thus, a string language L is called a HRS language if, up to treatment of the empty string, there is a HR grammar that generates the language of string graphs that represent exactly the strings in L.

The class of HRS languages is the union of all k-HRS languages for $k \in \mathbb{N}$.

Theorem (Precise Theorem 5)

For all $k \geq 1$, $\mathcal{HRS}_{2k} = \mathcal{HRS}_{2k+1} = \mathcal{MCF}_k$.

Parallel Derivations

Definition (Parallel Direct Derivation)

Let $H \in \mathcal{H}_{\mathcal{C}}$ with $E_H = \{e_1, \ldots e_n\}$, and $\mathcal{R}$ be a set of rules. If for each $e_i \in E_H$, there is an $R_i \in \mathcal{H}_{\mathcal{C}}$ such that $(\mathrm{lab}_H(e_i), R_i) \in \mathcal{R}$, then H parallelly directly derives $H' \cong H[e_1/R_1, \ldots, e_n/R_n]$, and write $H \Rrightarrow_{\mathcal{R}} H'$.

Parallel Derivations

Definition (Parallel Direct Derivation)

Let $H \in \mathcal{H}_{\mathcal{C}}$ with $E_H = \{e_1, \ldots e_n\}$, and $\mathcal{R}$ be a set of rules. If for each $e_i \in E_H$, there is an $R_i \in \mathcal{H}_{\mathcal{C}}$ such that $(\mathrm{lab}_H(e_i), R_i) \in \mathcal{R}$, then H parallelly directly derives $H' \cong H[e_1/R_1, \ldots, e_n/R_n]$, and write $H \Rrightarrow_{\mathcal{R}} H'$.

Definition (Parallel Derivation)

Let $\mathcal{S} = \{\mathcal{R}_i \mid i \in I\}$ be a finite set of rule sets indexed by I, and $\mathcal{M}$ an FSA over I. Then H ($\mathcal{M}$-)parallelly derives H' if there is a sequence $H \Rrightarrow_{\mathcal{R}_{i_1}} H_1 \Rrightarrow_{\mathcal{R}_{i_2}} \cdots \Rrightarrow_{\mathcal{R}_{i_k}} H_k \cong H'$ $(k \in \mathbb{N})$ such that $i_1 i_2 \cdots i_k \in \mathrm{L}(\mathcal{M})$. We write $H \Rrightarrow_{\mathcal{S}}^{\mathcal{M}} H'$, $H \Rrightarrow_{\mathcal{S}}^{i_1 i_2 \cdots i_k} H'$ or $H \Rrightarrow_{\mathcal{S}}^{k} H'$.

Parallel Derivations

Definition (Parallel Direct Derivation)

Let $H \in \mathcal{H}_\mathcal{C}$ with $E_H = \{e_1, \ldots e_n\}$, and $\mathcal{R}$ be a set of rules. If for each $e_i \in E_H$, there is an $R_i \in \mathcal{H}_\mathcal{C}$ such that $(\mathrm{lab}_H(e_i), R_i) \in \mathcal{R}$, then H parallelly directly derives $H' \cong H[e_1/R_1, \ldots, e_n/R_n]$, and write $H \Rrightarrow_\mathcal{R} H'$.

Definition (Parallel Derivation)

Let $\mathcal{S} = \{\mathcal{R}_i \mid i \in I\}$ be a finite set of rule sets indexed by I, and $\mathcal{M}$ an FSA over I. Then H ($\mathcal{M}$-)parallelly derives H' if there is a sequence $H \Rrightarrow_{\mathcal{R}_{i_1}} H_1 \Rrightarrow_{\mathcal{R}_{i_2}} \cdots \Rrightarrow_{\mathcal{R}_{i_k}} H_k \cong H'$ $(k \in \mathbb{N})$ such that $i_1 i_2 \cdots i_k \in \mathrm{L}(\mathcal{M})$. We write $H \Rrightarrow_\mathcal{S}^\mathcal{M} H'$, $H \Rrightarrow_\mathcal{S}^{i_1 i_2 \cdots i_k} H'$ or $H \Rrightarrow_\mathcal{S}^k H'$.

Definition (Table)

A table T is a finite set of rules over Σ such that for each $L \in \Sigma$, there is at least one $R \in \mathcal{H}_\mathcal{C}$ such that $(L, R) \in T$.

PHR Grammars I

Definition (PHR Grammar)

A (k-PHR grammar is a system $\mathcal{G} = (\mathcal{C}, A, S, \mathcal{T}, \mathcal{M})$ where:

1. $\mathcal{C} = (\Sigma, \mathrm{type})$ is a signature;
2. $A \subseteq \Sigma$ is the set of terminal labels;
3. $S \in \Sigma \setminus A$ is the start symbol;
4. $\mathcal{T} = \{T_i \mid i \in I\}$ is a finite set of tables indexed by I;
5. $\mathcal{M} = (Q, I, \delta, i, F)$ is an FSA over I;

with $\max(\{|type|(R) \mid (L, R) \in \bigcup_{T_i \in \mathcal{T}} T_i\}) \leq k$. The generated language is:

$$\mathrm{L}(\mathcal{G}) = \{H \in \mathcal{H}_{\mathcal{C}} \mid S^{\bullet} \Rrightarrow_{\mathcal{T}}^{\mathcal{M}} H \text{ with } \mathrm{lab}_H^{-1}(A) = E_H\} \subseteq \mathcal{H}_{\mathcal{C}}.$$

PHR Grammars I

Definition (PHR Grammar)

A (k-PHR grammar is a system $\mathcal{G} = (\mathcal{C}, A, S, \mathcal{T}, \mathcal{M})$ where:

1. $\mathcal{C} = (\Sigma, \mathrm{type})$ is a signature;
2. $A \subseteq \Sigma$ is the set of terminal labels;
3. $S \in \Sigma \setminus A$ is the start symbol;
4. $\mathcal{T} = \{T_i \mid i \in I\}$ is a finite set of tables indexed by I;
5. $\mathcal{M} = (Q, I, \delta, i, F)$ is an FSA over I;

with $\max(\{|type|(R) \mid (L, R) \in \bigcup_{T_i \in \mathcal{T}} T_i\}) \leq k$. The generated language is:

$$\mathrm{L}(\mathcal{G}) = \{H \in \mathcal{H}_{\mathcal{C}} \mid S^{\bullet} \Rrightarrow_{\mathcal{T}}^{\mathcal{M}} H \text{ with } \mathrm{lab}_H^{-1}(A) = E_H\} \subseteq \mathcal{H}_{\mathcal{C}}.$$

The inclusion of rational control, originally considered for ET0L by Asveld 1977, is a convenience to aid with the specification of human understandable grammars. This actually adds no power...

PHR Grammars II

Lemma (Control Removal)

Given a k-PHR grammar $\mathcal{G}$, one can effectively construct a k-PHR grammar $\mathcal{G}'$ without control such that $\mathrm{L}(\mathcal{G}) = \mathrm{L}(\mathcal{G}')$.

PHR Grammars II

Lemma (Control Removal)

Given a k-PHR grammar $\mathcal{G}$, one can effectively construct a k-PHR grammar $\mathcal{G}'$ without control such that $\mathrm{L}(\mathcal{G}) = \mathrm{L}(\mathcal{G}')$.

Proof: Encode control in the labels! Make a copy of all the labels for all of the states in the FSA, and moving between control states is synchronized with moving between the copies of labels. □

PHR Grammars II

Lemma (Control Removal)

Given a k-PHR grammar $\mathcal{G}$, one can effectively construct a k-PHR grammar $\mathcal{G}'$ without control such that $\mathrm{L}(\mathcal{G}) = \mathrm{L}(\mathcal{G}')$.

Proof: Encode control in the labels! Make a copy of all the labels for all of the states in the FSA, and moving between control states is synchronized with moving between the copies of labels. □

Proposition

$L = \{a^{2^n} \mid n \in \mathbb{N}\}$ is an ET0L language but not MCF or semilinear.

PHR Grammars II

Lemma (Control Removal)

Given a k-PHR grammar $\mathcal{G}$, one can effectively construct a k-PHR grammar $\mathcal{G}'$ without control such that $\mathrm{L}(\mathcal{G}) = \mathrm{L}(\mathcal{G}')$.

Proof: Encode control in the labels! Make a copy of all the labels for all of the states in the FSA, and moving between control states is synchronized with moving between the copies of labels. □

Proposition

$L = \{a^{2^n} \mid n \in \mathbb{N}\}$ is an ET0L language but not MCF or semilinear.

Theorem (PHR Generalises HR)

For $k \geq 0$, $\mathcal{HR}_k \subsetneq \mathcal{PHR}_k$.

PHR Grammars II

Lemma (Control Removal)

Given a k-PHR grammar $\mathcal{G}$, one can effectively construct a k-PHR grammar $\mathcal{G}'$ without control such that $\mathrm{L}(\mathcal{G}) = \mathrm{L}(\mathcal{G}')$.

Proof: Encode control in the labels! Make a copy of all the labels for all of the states in the FSA, and moving between control states is synchronized with moving between the copies of labels. □

Proposition

$L = \{a^{2^n} \mid n \in \mathbb{N}\}$ is an ET0L language but not MCF or semilinear.

Theorem (PHR Generalises HR)

For $k \geq 0$, $\mathcal{HR}_k \subsetneq \mathcal{PHR}_k$.

Proof: Induction on derivation length. □

PHRS Languages

Lemma (PHRS Generalises ET0L)

$\mathcal{ETOL} = \mathcal{PHRS}_2$ and for $k \geq 4$, $\mathcal{ETOL} \subsetneq \mathcal{PHRS}_k$.

PHRS Languages

Lemma (PHRS Generalises ET0L)

$\mathcal{ETOL} = \mathcal{PHRS}_2$ and for $k \geq 4$, $\mathcal{ETOL} \subsetneq \mathcal{PHRS}_k$.

Proof: Forward direction follows from by transforming 2-PHR such that all labels have at least type 2 and all RHS hypergraphs have proper edges. Reverse follows from transforming ET0L grammars to EPT0L. □

PHRS Languages

Lemma (PHRS Generalises ET0L)

$\mathcal{ETOL} = \mathcal{PHRS}_2$ and for $k \geq 4$, $\mathcal{ETOL} \subsetneq \mathcal{PHRS}_k$.

Proof: Forward direction follows from by transforming 2-PHR such that all labels have at least type 2 and all RHS hypergraphs have proper edges. Reverse follows from transforming ET0L grammars to EPT0L. □

Corollary

There are 2-PHRS languages that are not semilinear.

PHRS Languages

Lemma (PHRS Generalises ET0L)

$\mathcal{ETOL} = \mathcal{PHRS}_2$ and for $k \geq 4$, $\mathcal{ETOL} \subsetneq \mathcal{PHRS}_k$.

Proof: Forward direction follows from by transforming 2-PHR such that all labels have at least type 2 and all RHS hypergraphs have proper edges. Reverse follows from transforming ET0L grammars to EPT0L. □

Corollary

There are 2-PHRS languages that are not semilinear.

Lemma (PHRS Generalises MCF)

For $k \geq 2$, $\mathcal{HRS}_k \subsetneq \mathcal{PHRS}_k$.

PHRS Languages

Lemma (PHRS Generalises ET0L)

$\mathcal{ETOL} = \mathcal{PHRS}_2$ and for $k \geq 4$, $\mathcal{ETOL} \subsetneq \mathcal{PHRS}_k$.

Proof: Forward direction follows from by transforming 2-PHR such that all labels have at least type 2 and all RHS hypergraphs have proper edges. Reverse follows from transforming ET0L grammars to EPT0L. □

Corollary

There are 2-PHRS languages that are not semilinear.

Lemma (PHRS Generalises MCF)

For $k \geq 2$, $\mathcal{HRS}_k \subsetneq \mathcal{PHRS}_k$.

Proof: The equivalence of MCF and HRS due to Theorem 15. To see the remainder follows from Theorem 22 and its proof. We get strictness from Proposition 21 together with Lemma 23. □

Closing Conjectures

Conjecture (CS Generalises PHRS)

$\mathcal{PHRS} \subsetneq \mathcal{CS}$.

Closing Conjectures

Conjecture (CS Generalises PHRS)

$\mathcal{PHRS} \subsetneq \mathcal{CS}$.

Conjecture (Substitution-Closed Full AFL)

For $k \geq 2$, $\mathcal{PHRS}_k$ and $\mathcal{PHRS}$ are substitution-closed full AFLs.

Closing Conjectures

Conjecture (CS Generalises PHRS)

$\mathcal{PHRS} \subsetneq \mathcal{CS}$.

Conjecture (Substitution-Closed Full AFL)

For $k \geq 2$, $\mathcal{PHRS}_k$ and $\mathcal{PHRS}$ are substitution-closed full AFLs.

Conjecture (WP Double Torus)

The fundamental group of the double torus admits a PHRS word problem with is neither a MCF nor ET0L language.

Closing Conjectures

Conjecture (CS Generalises PHRS)

$\mathcal{PHRS} \subsetneq \mathcal{CS}$.

Conjecture (Substitution-Closed Full AFL)

For $k \geq 2$, $\mathcal{PHRS}_k$ and $\mathcal{PHRS}$ are substitution-closed full AFLs.

Conjecture (WP Double Torus)

The fundamental group of the double torus admits a PHRS word problem with is neither a MCF nor ET0L language.

Corollary

The word problem of any surface group is a PHRS language.

References I

Aho, Alfred (1968). "Indexed Grammars – An Extension of Context-Free Grammars". In: *Journal of the ACM* 15.4, pp. 647–671. DOI: 10.1145/321479.321488.

Anisimov, Anatoly (1971). "Group Languages". In: *Kibernetika* 4, pp. 18–24.

Asveld, Peter (1977). "Controlled iteration grammars and full hyper-AFL's". In: *Information and Control* 34.3, pp. 248–269. DOI: 10.1016/S0019-9958(77)90308-4.

Ciobanu, Laura, Murray Elder, and Michal Ferov (2018). "Applications of L systems to group theory". In: *International Journal of Algebra and Computation* 28.2, pp. 309–329. DOI: 10.1142/S0218196718500145.

Drewes, Frank, Hans-Jörg Kreowski, and Annegret Habel (1997). "Hyperedge Replacement Graph Grammars". In: *Handbook of Graph Grammars and Computing by Graph Transformation, Volume 1: Foundations*. Ed. by Grzegorz Rozenberg. World Scientific, pp. 95–162. DOI: 10.1142/9789812384720_0002.

Engelfriet, Joost and Linda Heyker (1991). "The string generating power of context-free hypergraph grammars". In: *Journal of Computer and System Sciences* 43.2, pp. 328–360. DOI: 10.1016/0022-0000(91)90018-Z.

Gilman, Robert, Robert Kropholler, and Saul Schleimer (2018). "Groups whose word problems are not semilinear". In: *Groups Complexity Cryptology* 10.2, pp. 53–62. DOI: 10.1515/gcc-2018-0010.

Habel, Annegret (1992). *Hyperedge Replacement: Grammars and Languages*. Vol. 643. Lecture Notes in Computer Science. Springer. DOI: 10.1007/BFb0013875.

Ho, Meng-Che (2018). "The word problem of $\mathbb{Z}^n$ is a multiple context-free language". In: *Groups Complexity Cryptology* 10.1, pp. 9–15. DOI: 10.1515/gcc-2018-0003.

References II

Kreowski, Hans-Jörg (1993). "Five facets of hyperedge replacement beyond context-freeness". In: *Proc. 9th International Conference on Fundamentals of Computation Theory (FCT 1993)*. Ed. by Zoltán Ésik. Vol. 710. Lecture Notes in Computer Science. Springer, pp. 69–86. DOI: 10.1007/3-540-57163-9_5.

Muller, David and Paul Schupp (1983). "Groups, the Theory of Ends, and Context-Free Languages". In: *Journal of Computer and System Sciences* 26.3, pp. 295–310. DOI: 10.1016/0022-0000(83)90003-X.

Novikov, Pyotr (1955). "Über die algorithmische Unentscheidbarkeit des Wortproblems in der Gruppentheorie". In: *Trudy Matematicheskogo Instituta imeni V.A. Steklova* 44, pp. 1–143.

Rozenberg, Grzegorz and Arto Salomaa (1980). *The Mathematical Theory of L Systems*. Vol. 90. Pure and Applied Mathematics. Academic Press.

Seki, Hiroyuki et al. (1991). "On multiple context-free grammars". In: *Theoretical Computer Science* 88.2, pp. 191–229. DOI: 10.1016/0304-3975(91)90374-B.

Weir, David (1992). "Linear context-free rewriting systems and deterministic tree-walking transducers". In: *Proc. 30th Annual Meeting of the Association for Computational Linguistics*. Ed. by Henry Thompson. Association for Computational Linguistics, pp. 136–143. DOI: 10.3115/981967.981985.