Multiple Context-free Grammars

Course 4: pumping properties

Sylvain Salvati

INRIA Bordeaux Sud-Ouest

ESSLLI 2011

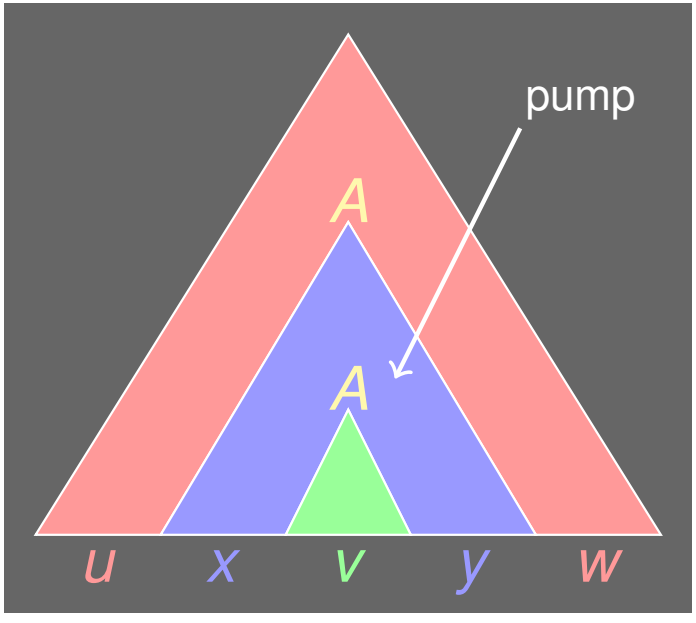
Outline

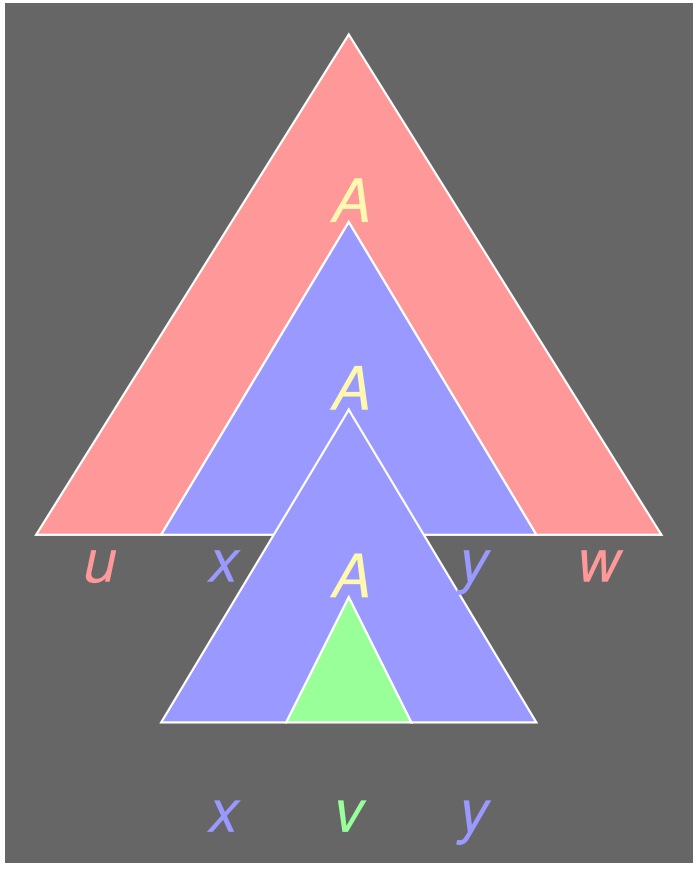
Ogden's pumping Lemma for CFL

- Ogden (1967)

COROLLARY. *For each context-free grammar* $G = (V, \Sigma, P, S)$ *there is an integer* k *such that any word* ξ *in* $L(G)$ *with* $|\xi| \geqslant k$ *has a decomposition of the form* $\xi = \alpha\beta\gamma\delta\mu$, *where* $|\beta\delta| > 0$, $|\beta\gamma\delta| \leqslant k$, *and for* $n \geqslant 0$, $\alpha\beta^n\gamma\delta^n\mu$ *is in* $L(G)$.

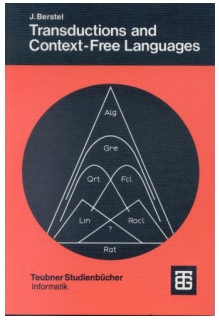
The pumping Lemma's proof





The pump is one of the *lowest* $\Rightarrow |xvy| < K$
The derivation is *minimal* in size $\Rightarrow xy \neq \epsilon$

A characterization of the recursive power of CFL



- Berstel (1979)

Theorem

Let $L \subseteq X^*$ be an algebraic language, and let π be a non-degenerated iterative pair in L. For any algebraic grammar generating L there exists an iterative pair $\bar{\pi}$ deduced from π and grammatical with respect to G.

Outline

Generalization of pumbability: k-pumpability

Definition (pumpability)

A language L is pumpable if there is K such that for all w such that $|w| > K$, $w = uxvyw$ with:

- $xy \neq \epsilon$
- for all $n \in \mathbb{N}$, $ux^n vy^n w$ is in L

Definition (k-pumpability)

A language L is k-pumpable if there is K such that for all w such that $|w| > K$, $w = v_1 x_1 v_2 x_2 \ldots v_k x_k v_{k+1}$ with:

- $x_1 \ldots x_k \neq \epsilon$
- for all $n \in \mathbb{N}$, $v_1 x_1^n v_2 x_2^n \ldots v_k x_k^n v_{k+1}$

- pumpability = 2-pumpability
- k-pumpability $\Rightarrow$ $k+1$-pumpability

Generalization of pumbability: k-pumpability

Definition (pumpability)

A language L is pumpable if there is K such that for all w such that $|w| > K$, $w = uxvyw$ with:

- $xy \neq \epsilon$
- for all $n \in \mathbb{N}$, $ux^n vy^n w$ is in L

Definition (k-pumpability)

A language L is k-pumpable if there is K such that for all w such that $|w| > K$, $w = v_1 x_1 v_2 x_2 \ldots v_k x_k v_{k+1}$ with:

- $x_1 \ldots x_k \neq \epsilon$
- for all $n \in \mathbb{N}$, $v_1 x_1^n v_2 x_2^n \ldots v_k x_k^n v_{k+1}$

- pumpability = 2-pumpability
- k-pumpability $\Rightarrow$ $k+1$-pumpability

Generalization of pumbability: k-pumpability

Definition (pumpability)

A language L is pumpable if there is K such that for all w such that $|w| > K$, $w = uxvyw$ with:

- $xy \neq \epsilon$
- for all $n \in \mathbb{N}$, $ux^n vy^n w$ is in L

Definition (k-pumpability)

A language L is k-pumpable if there is K such that for all w such that $|w| > K$, $w = v_1 x_1 v_2 x_2 \ldots v_k x_k v_{k+1}$ with:

- $x_1 \ldots x_k \neq \epsilon$
- for all $n \in \mathbb{N}$, $v_1 x_1^n v_2 x_2^n \ldots v_k x_k^n v_{k+1}$

- pumpability = 2-pumpability
- k-pumpability $\Rightarrow$ $k+1$-pumpability

Generalization of pumbability: k-pumpability

Definition (pumpability)

A language L is pumpable if there is K such that for all w such that $|w| > K$, $w = uxvyw$ with:

- $xy \neq \epsilon$
- for all $n \in \mathbb{N}$, $ux^n vy^n w$ is in L

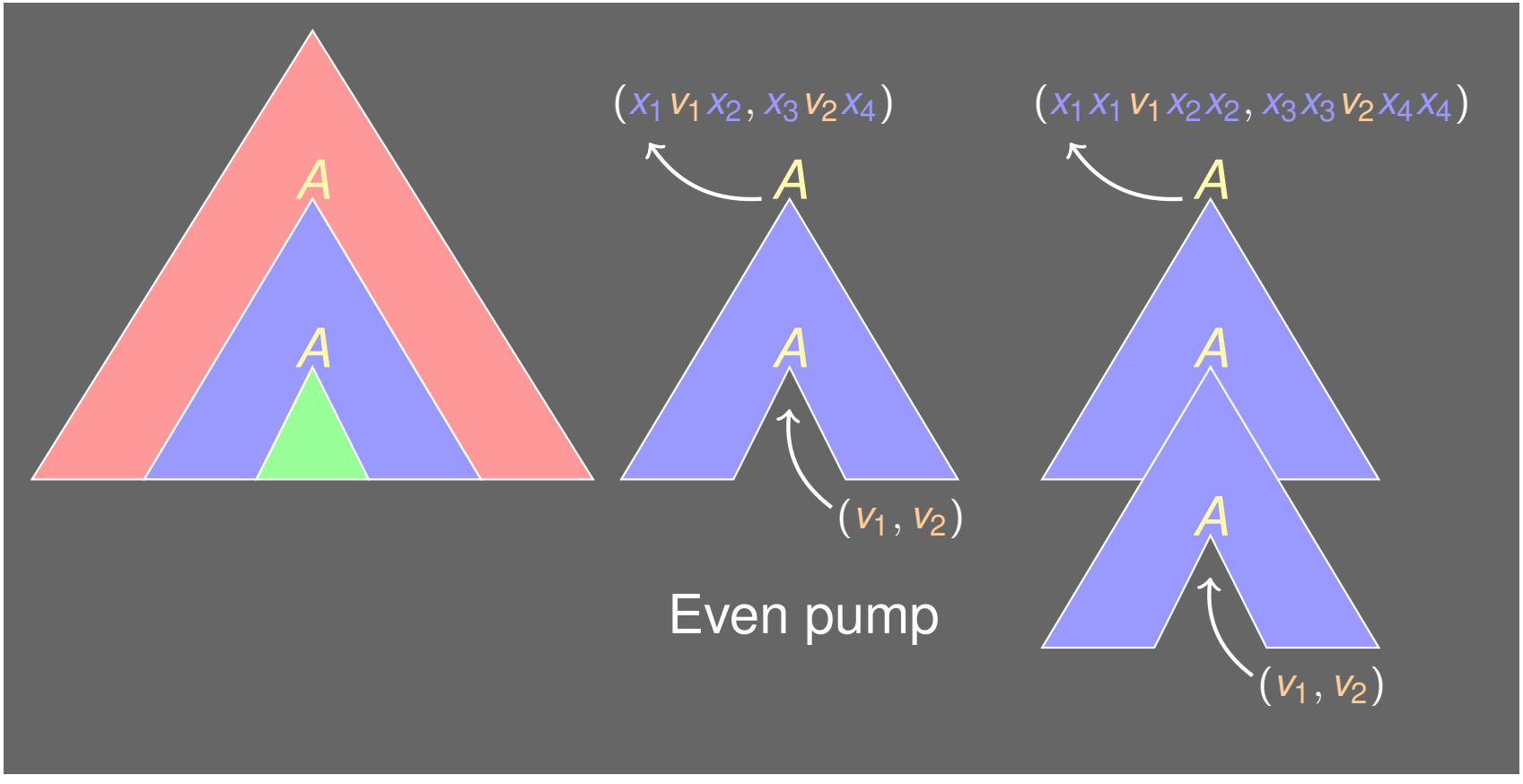
Definition (k-pumpability)

A language L is k-pumpable if there is K such that for all w such that $|w| > K$, $w = v_1 x_1 v_2 x_2 \ldots v_k x_k v_{k+1}$ with:

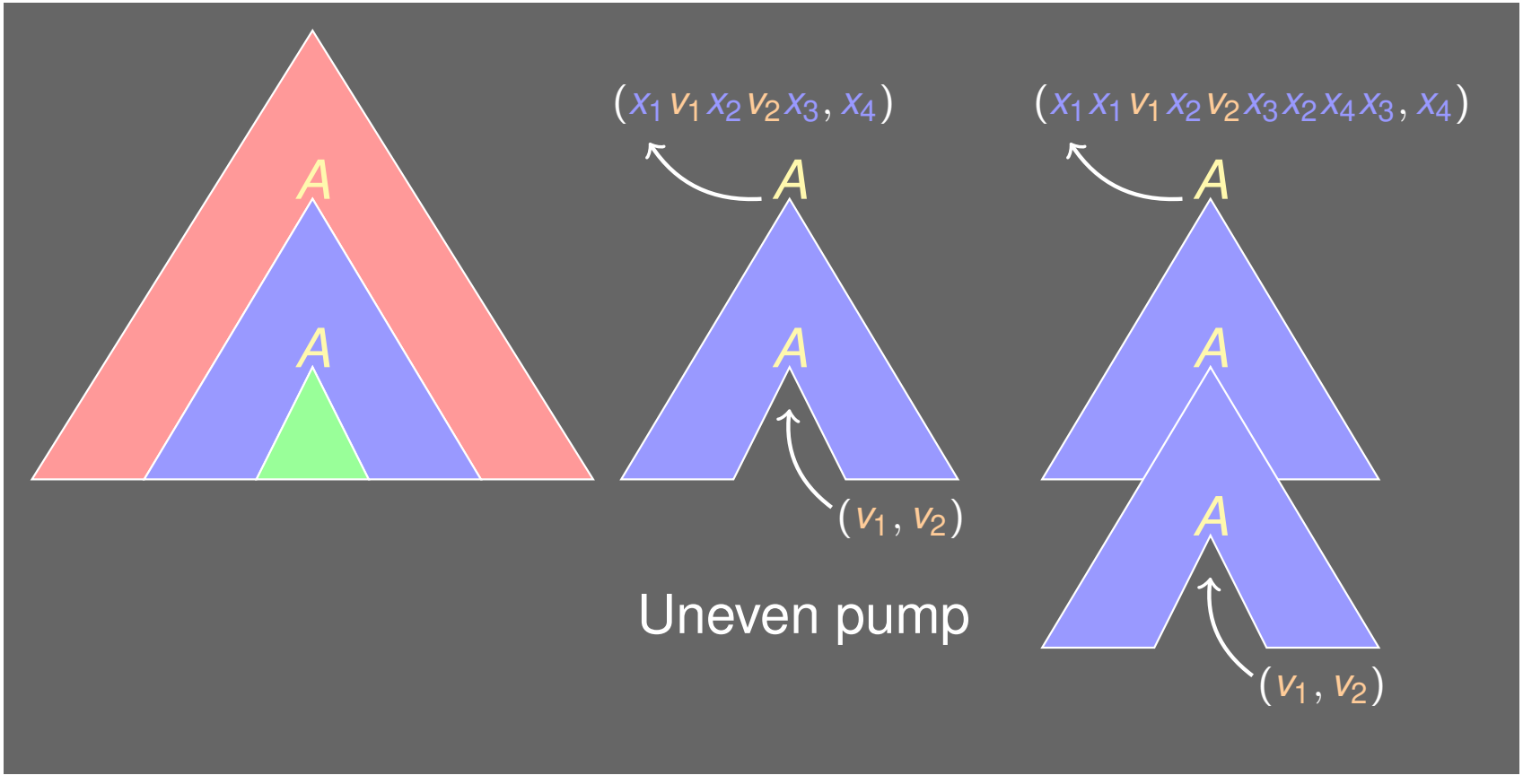
- $x_1 \ldots x_k \neq \epsilon$
- for all $n \in \mathbb{N}$, $v_1 x_1^n v_2 x_2^n \ldots v_k x_k^n v_{k+1}$

- pumpability = 2-pumpability
- k-pumpability $\Rightarrow$ $k+1$-pumpability

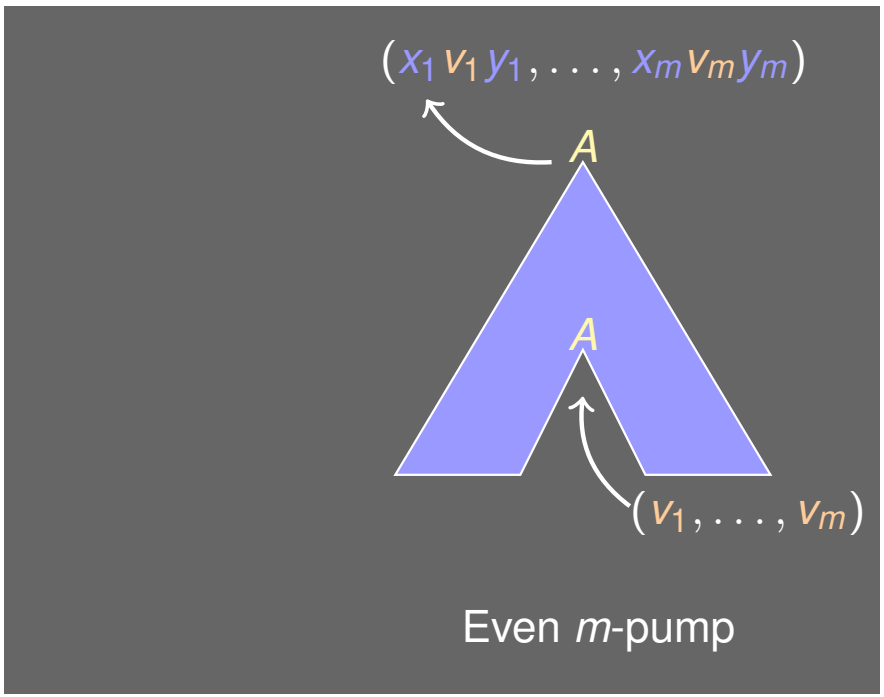
Iteration of the derivation tree



Iteration of the derivation tree



Even m-pump

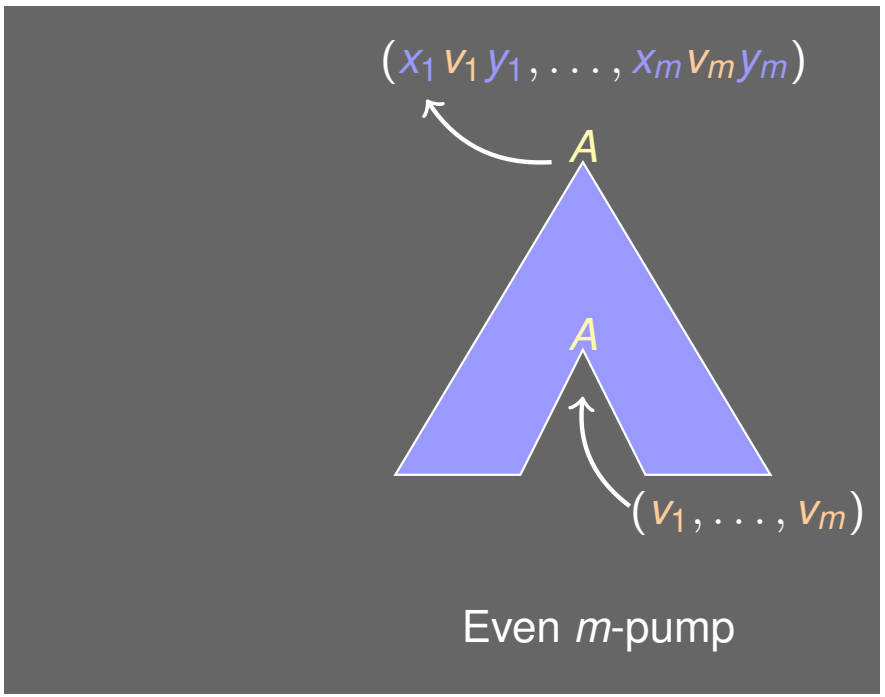


Even m-pump

Definition

A rule of a m-MCFG:
$A(s_1, \ldots, s_k) \leftarrow B_1(x_1^1, \ldots, x_{k_1}^1), \ldots, B_n(x_1^n, \ldots, x_{k_n}^n)$
is m-proper in the i^{th} premiss when $k = k_i = m$ and $s_j = w_1 x_j^i w_2$.

$A(y_1 x_1 y_2, x_2) \leftarrow B(x_1, x_2),\ C(y_1, y_2)$

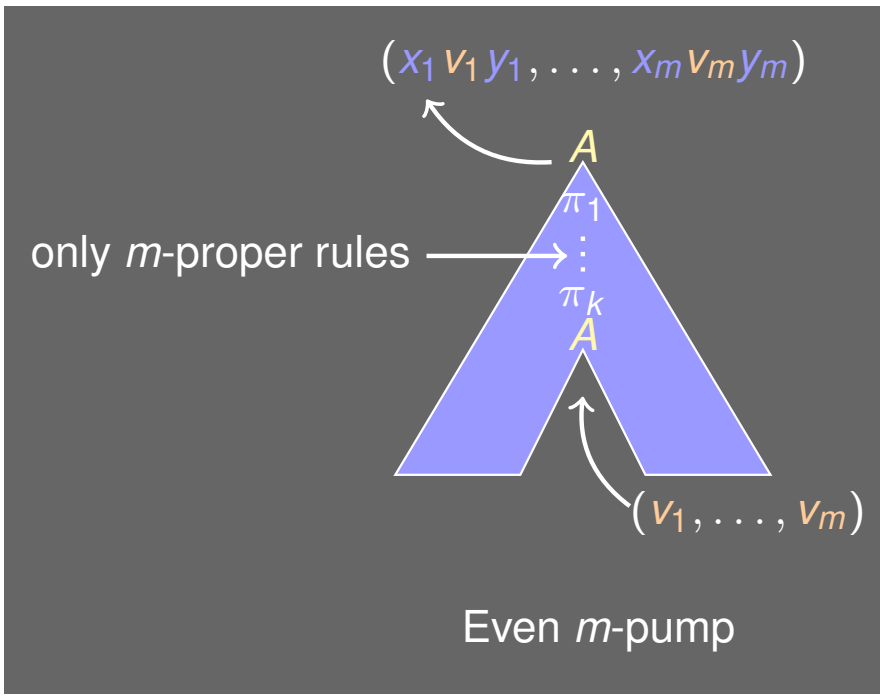


Even *m*-pump

Definition

A rule of a *m*-MCFG:
$A(s_1, \ldots, s_k) \leftarrow B_1(x_1^1, \ldots, x_{k_1}^1), \ldots, B_n(x_1^n, \ldots, x_{k_n}^n)$
is *m*-proper in the i^{th} premiss when $k = k_i = m$ and $s_j = w_1 x_j^i w_2$.

$A(y_1 x_1 y_2, x_2) \leftarrow B(x_1, x_2),\ C(y_1, y_2)$



Deviding the problem

- Strings that have a derivation tree containing an even m-pump are $2m$-pumpable.
- The set of derivation trees that contain an even m-pump is recognizable.
- From a m-MCFG$_{wn}$, G, we can construct two grammars G_1 and G_2, such that:
  - $\mathcal{L}(G) = \mathcal{L}(G_1) \cup \mathcal{L}(G_2)$
  - the derivation trees of G_1 all contain an even m-pump
  - no derivation tree of G_2 contains an even m-pump
- It remains to show that $\mathcal{L}(G_2)$ is $2m$-pumpable.

Deviding the problem

- Strings that have a derivation tree containing an even m-pump are $2m$-pumpable.
- The set of derivation trees that contain an even m-pump is recognizable.
- From a m-MCFG$_{wn}$, G, we can construct two grammars G_1 and G_2, such that:
  - $\mathcal{L}(G) = \mathcal{L}(G_1) \cup \mathcal{L}(G_2)$
  - the derivation trees of G_1 all contain an even m-pump
  - no derivation tree of G_2 contains an even m-pump
- It remains to show that $\mathcal{L}(G_2)$ is $2m$-pumpable.

Deviding the problem

- Strings that have a derivation tree containing an even m-pump are $2m$-pumpable.
- The set of derivation trees that contain an even m-pump is recognizable.
- From a m-MCFG$_{wn}$, G, we can construct two grammars G_1 and G_2, such that:
  - $\mathcal{L}(G) = \mathcal{L}(G_1) \cup \mathcal{L}(G_2)$
  - the derivation trees of G_1 all contain an even m-pump
  - no derivation tree of G_2 contains an even m-pump
- It remains to show that $\mathcal{L}(G_2)$ is $2m$-pumpable.

Deviding the problem

- Strings that have a derivation tree containing an even m-pump are $2m$-pumpable.
- The set of derivation trees that contain an even m-pump is recognizable.
- From a m-MCFG$_{wn}$, G, we can construct two grammars G_1 and G_2, such that:
  - $\mathcal{L}(G) = \mathcal{L}(G_1) \cup \mathcal{L}(G_2)$
  - the derivation trees of G_1 all contain an even m-pump
  - no derivation tree of G_2 contains an even m-pump
- It remains to show that $\mathcal{L}(G_2)$ is $2m$-pumpable.

Kanazawa's Lemma

- Kanazawa (2010)

Lemma

An m-MCFG$_{wn}$ whose derivation trees do not contain even m-pumps as an equivalent $m-1$-MCFG$_{wn}$.

Proof of the Lemma

Proof.

- eliminate m-proper rules by unfolding
- Make rules deriving non-terminal of arity m have no premise of arity m (relies on well-nestedness):

$$\frac{A(s_1,\ldots,s_m) \leftarrow B(x_1,\ldots,x_m),\ \Gamma}{\begin{array}{rcl} A(t_1,\ldots,t_m) & \leftarrow & D(y_1,\ldots,y_p),\ \Gamma_1 \\ D(u_1,\ldots,u_p) & \leftarrow & B(x_1,\ldots,x_m),\ \Gamma_2 \end{array}}$$

- Unfold the rules of non-terminal of arity m

The resulting grammar defines the same language and does not use non-terminals of arity m: it is an $m-1$-MCFG$_{wn}$ □

Thus by induction on m (the case where $m=1$ is the CFL case), $\mathcal{L}(G_2)$ is

$2(m-1)$-pumpable and therefore m-pumpable.

Proof of the Lemma

Proof.

- eliminate m-proper rules by unfolding
- Make rules deriving non-terminal of arity m have no premise of arity m (relies on well-nestedness):

$$\frac{\begin{array}{rcl} A(s_1,\ldots,s_m) & \leftarrow & B(x_1,\ldots,x_m),\ \Gamma \end{array}}{\begin{array}{rcl} A(t_1,\ldots,t_m) & \leftarrow & D(y_1,\ldots,y_p),\ \Gamma_1 \\ D(u_1,\ldots,u_p) & \leftarrow & B(x_1,\ldots,x_m),\ \Gamma_2 \end{array}}$$

- Unfold the rules of non-terminal of arity m

The resulting grammar defines the same language and does not use non-terminals of arity m: it is an $m-1$-MCFG$_{wn}$ □

Thus by induction on m (the case where $m=1$ is the CFL case), $\mathcal{L}(G_2)$ is

$2(m-1)$-pumpable and therefore m-pumpable.

Proof of the Lemma

Proof.

- eliminate m-proper rules by unfolding
- Make rules deriving non-terminal of arity m have no premise of arity m (relies on well-nestedness):

$$\frac{A(s_1,\ldots,s_m) \leftarrow B(x_1,\ldots,x_m),\ \Gamma}{\begin{array}{rcl} A(t_1,\ldots,t_m) & \leftarrow & D(y_1,\ldots,y_p),\ \Gamma_1 \\ D(u_1,\ldots,u_p) & \leftarrow & B(x_1,\ldots,x_m),\ \Gamma_2 \end{array}}$$

- Unfold the rules of non-terminal of arity m

The resulting grammar defines the same language and does not use non-terminals of arity m: it is an $m-1$-MCFG$_{wn}$ □

Thus by induction on m (the case where $m=1$ is the CFL case), $\mathcal{L}(G_2)$ is $2(m-1)$-pumpable and therefore m-pumpable.

Proof of the Lemma

Proof.

- eliminate m-proper rules by unfolding
- Make rules deriving non-terminal of arity m have no premise of arity m (relies on well-nestedness):

$$\frac{A(s_1,\ldots,s_m) \leftarrow B(x_1,\ldots,x_m),\ \Gamma}{\begin{array}{rcl} A(t_1,\ldots,t_m) & \leftarrow & D(y_1,\ldots,y_p),\ \Gamma_1 \\ D(u_1,\ldots,u_p) & \leftarrow & B(x_1,\ldots,x_m),\ \Gamma_2 \end{array}}$$

- Unfold the rules of non-terminal of arity m

The resulting grammar defines the same language and does not use non-terminals of arity m: it is an $m-1$-MCFG$_{wn}$ □

Thus by induction on m (the case where $m=1$ is the CFL case), $\mathcal{L}(G_2)$ is $2(m-1)$-pumpable and therefore m-pumpable.

Proof of the Lemma

Proof.

- eliminate m-proper rules by unfolding
- Make rules deriving non-terminal of arity m have no premise of arity m (relies on well-nestedness):

$$\frac{A(s_1, \ldots, s_m) \leftarrow B(x_1, \ldots, x_m), \Gamma}{\begin{array}{rcl} A(t_1, \ldots, t_m) & \leftarrow & D(y_1, \ldots, y_p), \Gamma_1 \\ D(u_1, \ldots, u_p) & \leftarrow & B(x_1, \ldots, x_m), \Gamma_2 \end{array}}$$

- Unfold the rules of non-terminal of arity m

The resulting grammar defines the same language and does not use non-terminals of arity m: it is an $m-1$-MCFG$_{wn}$ □

Thus by induction on m (the case where $m = 1$ is the CFL case), $\mathcal{L}(G_2)$ is $2(m-1)$-pumpable and therefore m-pumpable.

Proof of the Lemma

Proof.

- eliminate m-proper rules by unfolding
- Make rules deriving non-terminal of arity m have no premise of arity m (relies on well-nestedness):

$$\frac{A(s_1,\ldots,s_m) \leftarrow B(x_1,\ldots,x_m),\ \Gamma}{\begin{array}{rcl} A(t_1,\ldots,t_m) & \leftarrow & D(y_1,\ldots,y_p),\ \Gamma_1 \\ D(u_1,\ldots,u_p) & \leftarrow & B(x_1,\ldots,x_m),\ \Gamma_2 \end{array}}$$

- Unfold the rules of non-terminal of arity m

The resulting grammar defines the same language and does not use non-terminals of arity m: it is an $m-1$-MCFG$_{wn}$ □

Thus by induction on m (the case where $m = 1$ is the CFL case), $\mathcal{L}(G_2)$ is

$2(m-1)$-pumpable and therefore m-pumpable.

Kanazawa's Theorems

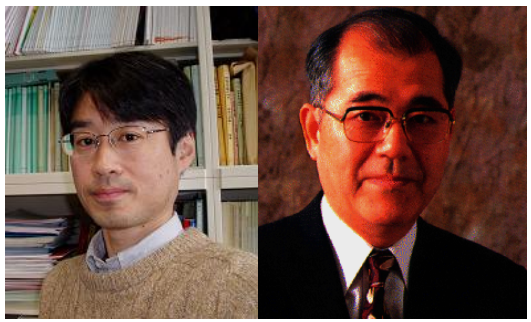
- Kanazawa (2010)

Theorem
m-MCFL$_{wn}$ are 2m-pumpable.

Theorem
2-MCFL are 4-pumpable.

Outline

Seki *et al.* pumping lemma

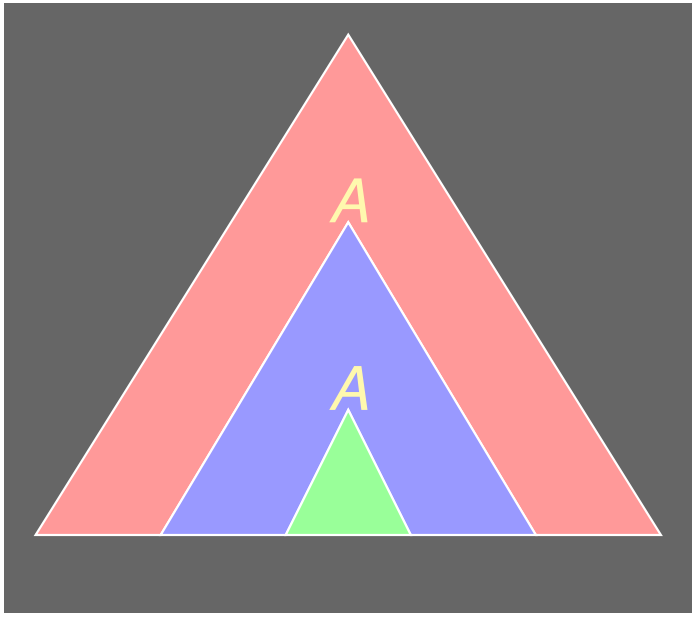


- Seki, Matsumura, Fujii, Kasami

Theorem

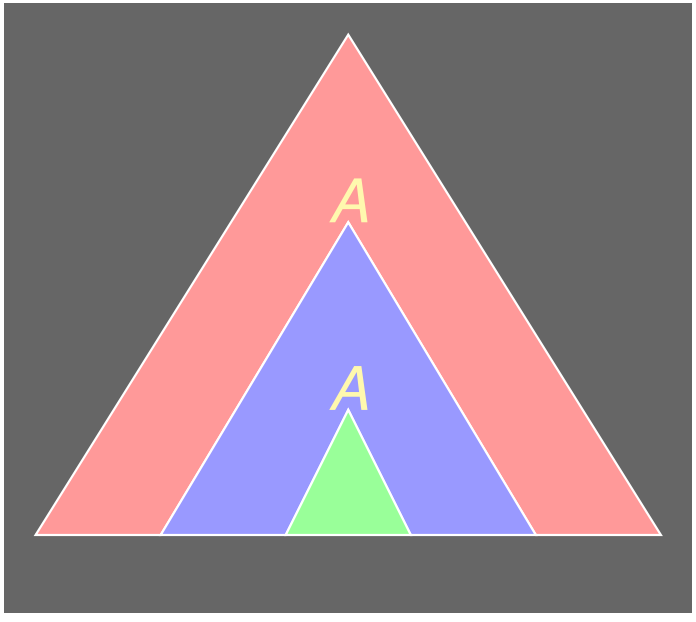
In a infinite m-MCFL L, there is a string w that is 2m-pumpable in L.

Proof of the Theorem



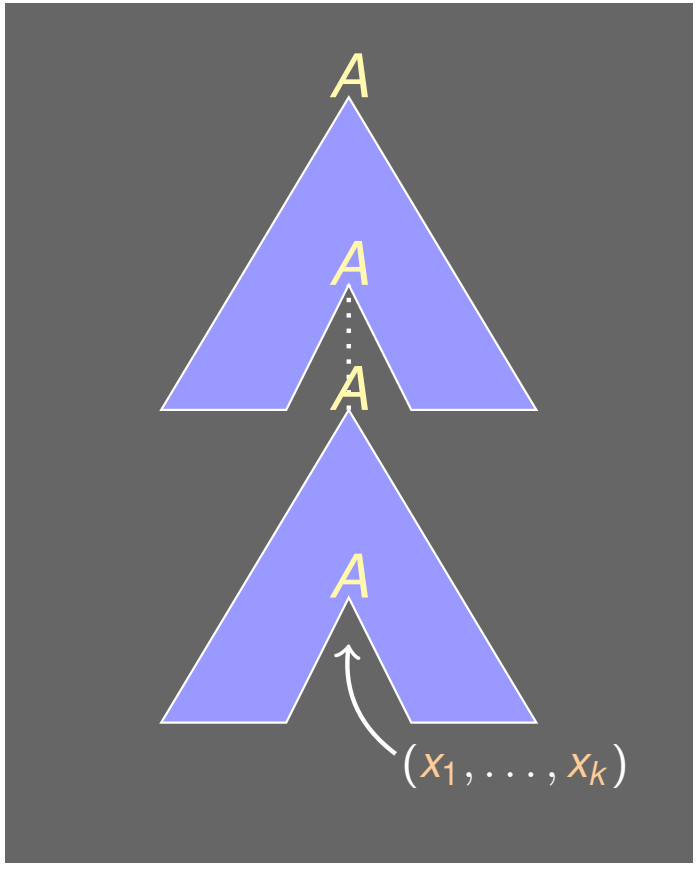
Take an ordered grammar.

Proof of the Theorem

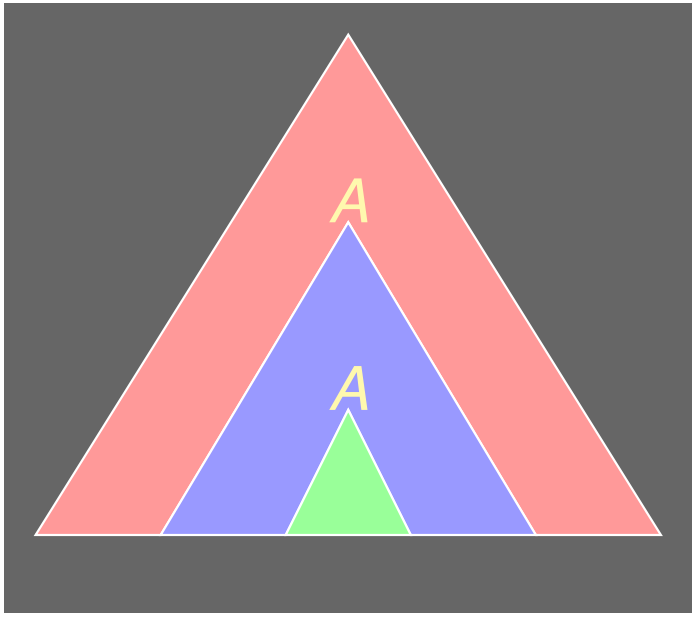


Take an ordered grammar.

Iterate the pump until the variables are fixed.



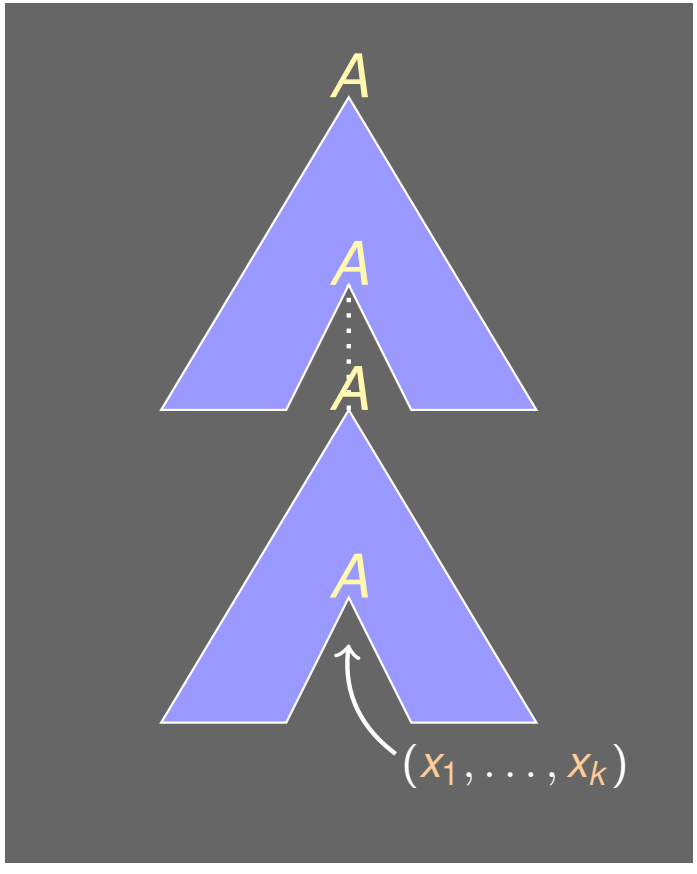
Proof of the Theorem



Take an ordered grammar.

Iterate the pump until the variables are fixed.

Now iterating the pump results in an iteration on the string.



An example

The pump:

$$(u_1 x_1 u_2 x_2 u_3, u_4)$$

One iteration:

$$(u_1 \underline{u_1 x_1 u_2 x_2 u_3} u_2 \underline{u_4} u_3, u_4)$$

The variables are fixed.
Second iteration:

$$(u_1 \underline{u_1 u_1 x_1 u_2 x_2 u_3 u_2 u_4 u_3} \underline{u_2 u_4} u_3, u_4)$$

An example

The pump:

$$(u_1 x_1 u_2 x_2 u_3, u_4)$$

One iteration:

$$(u_1 \underline{u_1 x_1 u_2 x_2 u_3} u_2 \underline{u_4} u_3, u_4)$$

The variables are fixed.
Second iteration:

$$(u_1 \underline{u_1 u_1 x_1 u_2 x_2 u_3 u_2 u_4 u_3} \underline{u_2 u_4} u_3, u_4)$$

An example

The pump:

$$(u_1 x_1 u_2 x_2 u_3, u_4)$$

One iteration:

$$(u_1 \underline{u_1 x_1 u_2 x_2 u_3} u_2 \underline{u_4} u_3, u_4)$$

The variables are fixed.

Second iteration:

$$(u_1 \underline{u_1 u_1 x_1 u_2 x_2 u_3 u_2 u_4 u_3} \underline{u_2 u_4} u_3, u_4)$$

An example

The pump:

$$(u_1 x_1 u_2 x_2 u_3, u_4)$$

One iteration:

$$(u_1 \underline{u_1 x_1 u_2 x_2 u_3} u_2 \underline{u_4} u_3, u_4)$$

The variables are fixed.
Second iteration:

$$(u_1 \ \underline{u_1 u_1 x_1 u_2 x_2 u_3 u_2 u_4 u_3} \ u_2 \underline{u_4} u_3, u_4)$$

An example

The pump:

$$(u_1 x_1 u_2 x_2 u_3, u_4)$$

One iteration:

$$(u_1 \underline{u_1 x_1 u_2 x_2 u_3} u_2 \underline{u_4} u_3, u_4)$$

The variables are fixed.
Second iteration:

$$(\underline{u_1}\ \underline{u_1} u_1 x_1 u_2 x_2 u_3 \underline{u_2 u_4 u_3}\ \underline{u_2 u_4 u_3}, u_4)$$

Outline

A mythic pumping Lemma

- Radzinski (1991)

Pumping Lemma for MCTALs

If L is a MCTAL, then there are constants n and m such that if $z \in \mathrm{L}$ and $|z| \geq n$ then z may be written as $z = u_1 v_1 w_1 s_1 u_2 v_2 w_2 s_2 u_3 \ldots u_m v_m w_m s_m u_{m+1}$ with $\sum_{j=1}^{m} |v_j s_j| \geq 1$ such that for all $i \geq 0$, $u_1 v_1^i w_1 s_1^i u_2 v_2^i w_2 s_2^i u_3 \ldots u_m v_m^i w_m s_m^i u_{m+1} \in \mathrm{L}$.

I has then been assumed to hold: Groenink 1997, Kracht 2003

Constructing a counter-example

- We must take a m-MCFL so that $m > 2$
- It must not contain proper rules

$$
\begin{aligned}
H(x_2) &\leftarrow G(x_1, x_2, x_3) \\
G(ax_1, y_1 c x_2 \bar{c} d y_2 \bar{d} x_3, y_3 b) &\leftarrow G(x_1, x_2, x_3)\, G(y_1, y_2, y_3) \\
G(a, \epsilon, b) &\leftarrow
\end{aligned}
$$

The language and binary trees

$$\begin{aligned} H(x_2) &\leftarrow G(x_1, x_2, x_3) \\ G(ax_1, y_1 c x_2 \overline{c} d y_2 \overline{d} x_3, y_3 b) &\leftarrow G(x_1, x_2, x_3)\, G(y_1, y_2, y_3) \\ G(a, \epsilon, b) &\leftarrow \end{aligned}$$

Let φ be a morphism such that $\varphi(a) = \varphi(b) = \epsilon$ and leaves the other letter unchanged.

Lemma

$\varphi(\mathcal{L}(H))$ is the CFL described by the grammar $V \to cV\overline{c}dV\overline{d} \mid \epsilon$

The language $\mathcal{L}(V)$ represent binary trees.
The language $\mathcal{L}(H)$ can be seen as:

The language and binary trees

$$\begin{aligned} H(x_2) &\leftarrow G(x_1, x_2, x_3) \\ G(ax_1, y_1 c x_2 \overline{c} d y_2 \overline{d} x_3, y_3 b) &\leftarrow G(x_1, x_2, x_3)\, G(y_1, y_2, y_3) \\ G(a, \epsilon, b) &\leftarrow \end{aligned}$$

Let φ be a morphism such that $\varphi(a) = \varphi(b) = \epsilon$ and leaves the other letter unchanged.

Lemma

$\varphi(\mathcal{L}(H))$ is the CFL described by the grammar $V \to cV\overline{c}dV\overline{d} \mid \epsilon$

The language $\mathcal{L}(V)$ represent binary trees.

The language $\mathcal{L}(H)$ can be seen as:

The language and binary trees

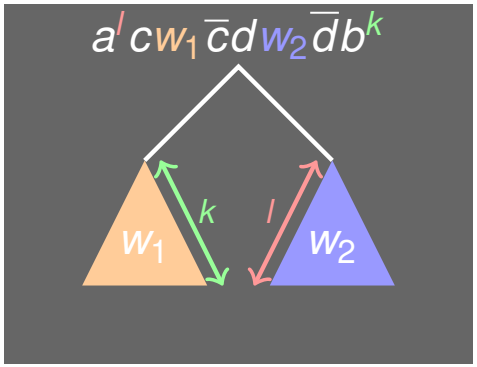
$$\begin{aligned} H(x_2) &\leftarrow G(x_1, x_2, x_3) \\ G(ax_1, y_1 c x_2 \overline{c} d y_2 \overline{d} x_3, y_3 b) &\leftarrow G(x_1, x_2, x_3)\, G(y_1, y_2, y_3) \\ G(a, \epsilon, b) &\leftarrow \end{aligned}$$

Let φ be a morphism such that $\varphi(a) = \varphi(b) = \epsilon$ and leaves the other letter unchanged.

Lemma

$\varphi(\mathcal{L}(H))$ is the CFL described by the grammar $V \to cV\overline{c}dV\overline{d} \mid \epsilon$

The language $\mathcal{L}(V)$ represent binary trees.
The language $\mathcal{L}(H)$ can be seen as:



3-MCFL are not finitely pumpable

- Kanazawa, Kobele, S., Yoshinaka 2011



Theorem

$\mathcal{L}(H)$ is not k-pumpable for any k

Proof.

Every string representing complete binary tree is not k-pumpable in $\mathcal{L}(H)$ for any k. □

Outline

MCFL$_{wn}$ and cross serial dependencies

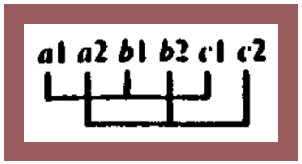
- Joshi's informal notion

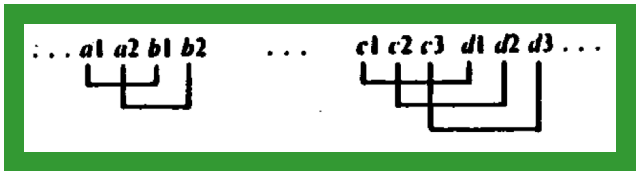
$I(x_1y_1, y_2x_2) \leftarrow J(x_1, x_2),\ K(y_1, y_2)$

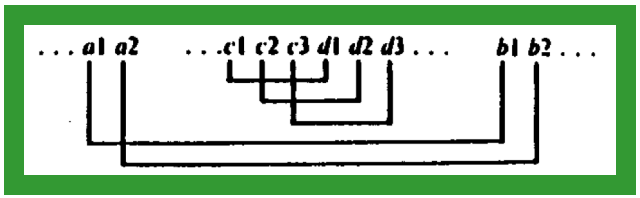
$I(x_1y_1, x_2y_2) \leftarrow J(x_1, x_2),\ K(y_1, y_2)$

$A(x_1z_1, z_2x_2y_1, y_2y_3x_3) \leftarrow B(x_1, x_2, x_3)\ C(y_1, y_2, y_3)\ D(z_1, z_2)$

$A(z_1x_1, y_1x_2z_2y_2x_3, y_3) \leftarrow B(x_1, x_2, x_3)\ C(y_1, y_2, y_3)\ D(z_1, z_2)$







MCFL$_{wn}$ and cross serial dependencies

- Joshi, Vijay Shanker, Weir (1991)

in MCSL; 2) languages in MCSL can be parsed in polynomial time; 3) MCSGs capture only certain kinds of dependencies, e.g., nested dependencies and certain limited kinds of crossing dependencies (e.g., in the subordinate clause constructions in Dutch or some variations of them, but perhaps not in the so-called MIX (or Bach) language, which consists of equal numbers of a's, b's, and c's in any order 4) languages in MCSL have constant growth property, i.e., if the strings of a language

Conjecture: MIX is not a MCFL$_{wn}$

MCFL$_{wn}$ and cross serial dependencies

- Groenink (PhD dissertation): *finite pumpability*

8-8 deÆnition: Ænite pumpability. A language L is FINITELY PUMPABLE if there is a constant c_0 such that for any $w \in L$ with $|w| > c_0$, there are a finite number k and strings $u_0, \ldots, u_k$ and $v_1, \ldots, v_k$ such that $w = u_0 v_1 u_1 v_2 u_2 \cdots u_{k-1} v_k u_k$, and for each $i, 1 \leq |v_i| < c_0$, and for any $p \geq 0$, $u_0 v_1^p u_1 v_2^p u_2 \cdots u_{k-1} v_k^p u_k \in L$.

k-MCFL$_{wn}$ are $2k$-pumpable.

MCFL$_{wn}$ and cross serial dependencies

- Kallmeyer (course Düsseldorf): *finite copying*

Cross-Serial Dependencies

The second property (limited amount of cross-serial dependencies) is a little unclear. It can be taken to mean the following:

There is an $n \geq 2$ such that $\{w^k \mid w \in T^*\} \in \mathcal{L}$ for all $k \leq n$.

For every $k \in \mathbb{N}$, $\{w^k \mid w \in T^*\}$ is a k-MCFL$_{wn}$.

MCFL$_{wn}$ and constant growth property

- Vijay Shanker, Weir, Joshi (1987)

Theorem

The language defined by an MCFG$_{wn}$ is semilinear.

MCFL$_{wn}$ Polynomial parsing

	Membership	Universal Membership
m-MCFG	LOGCFL-complete	NP-complete (when $m \geq 2$)
m-MCFG$_{wn}$	LOGCFL-complete	P-complete
MCFG	LOGCFL-complete	PSPACE-complete/ EXPTIME-complete
MCFG$_{wn}$	LOGCFL-complete	PSPACE-complete/ ??