Review for CMSC 452 Midterm: Grammars

Context Free Languages

Examples of Context Free Grammars

$S \rightarrow aSb$
$S \rightarrow e$

The set of all strings **Generated** is

Examples of Context Free Grammars

$S \to aSb$
$S \to e$

The set of all strings **Generated** is

$$L = \{a^n b^n : n \in \mathbb{N}\}$$

Examples of Context Free Grammars

$S \to aSb$
$S \to e$

The set of all strings **Generated** is

$$L = \{a^n b^n : n \in \mathbb{N}\}$$

Note L is context free lang that is not regular.

Context Free Grammar for $\{a^{2n}b^n : n \in \mathbb{N}\}$

$S \rightarrow aaSb$
$S \rightarrow e$

The set of all strings **Generated** is

Context Free Grammar for $\{a^{2n}b^n : n \in \mathbb{N}\}$

$S \to aaSb$
$S \to e$

The set of all strings **Generated** is

$$L = \{a^{2n}b^n : n \in \mathbb{N}\}$$

Context Free Grammar for $\{a^{2n}b^n : n \in \mathbb{N}\}$

$S \rightarrow aaSb$
$S \rightarrow e$

The set of all strings **Generated** is

$$L = \{a^{2n}b^n : n \in \mathbb{N}\}$$

Note L is context free lang that is not regular.

Context Free Grammar for $\{a^m b^n : m > n\}$

Context Free Grammar for $\{a^m b^n : m > n\}$

$S \rightarrow AT$
$T \rightarrow aTb$
$T \rightarrow e$
$A \rightarrow Aa$
$A \rightarrow a$

Context Free Grammars

Def A **Context Free Grammar** is a tuple $G = (N, \Sigma, R, S)$

- N is a finite set of **nonterminals**.
- Σ is a finite **alphabet**. Note $\Sigma \cap N = \emptyset$.
- $R \subseteq N \times (N \cup \Sigma)^*$ and are called **Rules**.
- $S \in N$, the **start symbol**.

L(G)

If A is non-terminal then the CFG gives us gives us rules like:

- $A \rightarrow AB$
- $A \rightarrow a$

L(G)

If A is non-terminal then the CFG gives us gives us rules like:

- $A \to AB$
- $A \to a$

For any string of **terminals and non-terminals** α, $A \Rightarrow \alpha$ means that, starting from A, some combination of the rules produces α.

L(G)

If A is non-terminal then the CFG gives us gives us rules like:

- $A \rightarrow AB$
- $A \rightarrow a$

For any string of **terminals and non-terminals** α, $A \Rightarrow \alpha$ means that, starting from A, some combination of the rules produces α.

Examples:

- $A \Rightarrow a$
- $A \Rightarrow aB$

L(G)

If A is non-terminal then the CFG gives us gives us rules like:

- $A \to AB$
- $A \to a$

For any string of **terminals and non-terminals** α, $A \Rightarrow \alpha$ means that, starting from A, some combination of the rules produces α.

Examples:

- $A \Rightarrow a$
- $A \Rightarrow aB$

Then, if w is string of **terminals only**, we define $L(G)$ by:

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow w\}$$

Number of a's = Number of b's

Is

$$L = \{w \mid \#_a(w) = \#_b(w)\}$$

context free?

YES

Let G be the CFG

$S \to aSb$

$S \to bSa$

$S \to SS$

$S \to e$

YES

Let G be the CFG

$S \to aSb$

$S \to bSa$

$S \to SS$

$S \to e$

Thm $L(G) = \{w \mid \#_a(w) = \#_b(w)\}$.

YES

Let G be the CFG
$S \rightarrow aSb$
$S \rightarrow bSa$
$S \rightarrow SS$
$S \rightarrow e$

Thm $L(G) = \{w \mid \#_a(w) = \#_b(w)\}$.

Note This Theorem is **not obvious**. Deserves a proof! But I won't give one.

Example of a Lang that is NOT a CFL

1) $\{a^n b^n c^n : n \in \mathbb{N}\}$ is NOT a CFL.

Example of a Lang that is NOT a CFL

1) $\{a^n b^n c^n : n \in \mathbb{N}\}$ is NOT a CFL.

2) $\{a^{n^2} : n \in \mathbb{N}\}$ is NOT a CFL.

Example of a Lang that is NOT a CFL

1) $\{a^n b^n c^n : n \in \mathbb{N}\}$ is NOT a CFL.

2) $\{a^{n^2} : n \in \mathbb{N}\}$ is NOT a CFL.

3) If $L \subseteq a^*$ and L is not regular than L is not a CFL.

Example of a Lang that is NOT a CFL

1) $\{a^n b^n c^n : n \in \mathbb{N}\}$ is NOT a CFL.

2) $\{a^{n^2} : n \in \mathbb{N}\}$ is NOT a CFL.

3) If $L \subseteq a^*$ and L is not regular than L is not a CFL.

One proves theorems NON CFL using the PL for CFL's (next slide).

Closure Properties and REG$\subset$ CFL

L_1, L_2 CFL $\rightarrow$ $L_1 \cup L_2$ CFL

L_1 is CFL via CFG (N_1, Σ, R_1, S_1).
L_2 is CFL via CFG (N_2, Σ, R_2, S_2).

L_1, L_2 CFL $\rightarrow L_1 \cup L_2$ CFL

L_1 is CFL via CFG (N_1, Σ, R_1, S_1).
L_2 is CFL via CFG (N_2, Σ, R_2, S_2).

CFL for $L_1 \cup L_2$:
Just add $S \rightarrow S_1$ and $S \rightarrow S_2$ to union of grammars.

L_1, L_2 CFL $\rightarrow L_1 \cap L_2$ CFL

NOT TRUE: $a^n b^n c^* \cap a^* b^n c^n = a^n b^n c^n$.

L_1, L_2 CFL $\rightarrow$ $L_1 \cdot L_2$ CFL

L_1 is CFL via CFG (N_1, Σ, R_1, S_1).
L_2 is CFL via CFG (N_2, Σ, R_2, S_2).

L_1, L_2 CFL $\rightarrow L_1 \cdot L_2$ CFL

L_1 is CFL via CFG (N_1, Σ, R_1, S_1).
L_2 is CFL via CFG (N_2, Σ, R_2, S_2).

CFL for $L_1 \cup L_2$:
Just add $S \rightarrow S_1 S_2$ to union of grammars.

L CFL $\rightarrow \overline{L}$ CFL

FALSE.
Let

$$L = \overline{\{a^n b^n c^n : n \in \mathbb{N}\}}$$

L CFL $\rightarrow \overline{L}$ CFL

FALSE.
Let

$$L = \overline{\{a^n b^n c^n : n \in \mathbb{N}\}}$$

This is a CFL. This will a HW.

L CFL $\rightarrow L^*$ CFL

L is CFL via CFG (N, Σ, R, S).

L CFL $\to L^*$ CFL

L is CFL via CFG (N, Σ, R, S).

This one I leave to you to look up my slides on it.

REG contained in CFL

For every **regex** α, $L(\alpha)$ is a CFL.

REG contained in CFL

For every **regex** α, $L(\alpha)$ is a CFL.

Prove by ind on the length of α.

REG contained in CFL

For every **regex** α, $L(\alpha)$ is a CFL.

Prove by ind on the length of α.

We omit from this review.

REG contained in CFL

For every DFA M $L(M)$ is a CFL.

REG contained in CFL

For every DFA M $L(M)$ is a CFL.

Prove by construction. You did this on the HW.

REG contained in CFL

For every DFA M $L(M)$ is a CFL.

Prove by construction. You did this on the HW.

We omit from this review.

Examples of CFL's and Size of CFG's

Chomsky Normal Form

Def CFG G is in **Chomsky Normal Form** if the rules are all of the following form:

Chomsky Normal Form

Def CFG G is in **Chomsky Normal Form** if the rules are all of the following form:
1) $A \to BC$ where $A, B, C \in N$ (nonterminals).

Chomsky Normal Form

Def CFG G is in **Chomsky Normal Form** if the rules are all of the following form:
1) $A \rightarrow BC$ where $A, B, C \in N$ (nonterminals).
2) $A \rightarrow \sigma$ (where $A \in N$ and $\sigma \in \Sigma$).

Chomsky Normal Form

Def CFG G is in **Chomsky Normal Form** if the rules are all of the following form:
1) $A \rightarrow BC$ where $A, B, C \in N$ (nonterminals).
2) $A \rightarrow \sigma$ (where $A \in N$ and $\sigma \in \Sigma$).
3) $S \rightarrow e$ (where S is the start state).

Example of Chomsky Normal Form

Chomsky Normal form CFG that generates $\{aaaaaaaa\}$
$S \rightarrow AA$

Example of Chomsky Normal Form

Chomsky Normal form CFG that generates $\{aaaaaaaa\}$

$S \to AA$

$A \to BB$

Example of Chomsky Normal Form

Chomsky Normal form CFG that generates $\{aaaaaaaa\}$

$S \rightarrow AA$

$A \rightarrow BB$

$B \rightarrow CC$

Example of Chomsky Normal Form

Chomsky Normal form CFG that generates $\{aaaaaaaa\}$

$S \rightarrow AA$

$A \rightarrow BB$

$B \rightarrow CC$

$C \rightarrow a$

Example of Chomsky Normal Form

Chomsky Normal form CFG that generates $\{aaaaaaaa\}$
$S \to AA$
$A \to BB$
$B \to CC$
$C \to a$
So $\{aaaaaaaa\}$ has a CFG of size 4.

Example of Chomsky Normal Form

Chomsky Normal form CFG that generates $\{aaaaaaaa\}$
$S \to AA$
$A \to BB$
$B \to CC$
$C \to a$
So $\{aaaaaaaa\}$ has a CFG of size 4.

By the same trick $\exists$ a CFG for $\{a^n\}$ of size $O(\log n)$.

- Any DFA or NFA that recognizes $\{a^n\}$ has $n + \Omega(1)$ states.
- There is a CFG that generates $\{a^n\}$ with $O(\log n)$ rules.

Example of Chomsky Normal Form

Chomsky Normal form CFG that generates $\{aaaaaaaa\}$
$S \to AA$
$A \to BB$
$B \to CC$
$C \to a$
So $\{aaaaaaaa\}$ has a CFG of size 4.

By the same trick $\exists$ a CFG for $\{a^n\}$ of size $O(\log n)$.

- Any DFA or NFA that recognizes $\{a^n\}$ has $n + \Omega(1)$ states.
- There is a CFG that generates $\{a^n\}$ with $O(\log n)$ rules.

This only worked so well since a^n is a very simple string.

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

$S \to [A][ABBBAB]$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

$S \rightarrow [A][ABBBAB]$

$[ABBBAB] \rightarrow [A][BBBAB]$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

$S \to [A][ABBBAB]$

$[ABBBAB] \to [A][BBBAB]$

$[BBBAB] \to [B][BBAB]$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

$S \to [A][ABBBAB]$

$[ABBBAB] \to [A][BBBAB]$

$[BBBAB] \to [B][BBAB]$

$[BBAB] \to [B][BAB]$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

$S \to [A][ABBBAB]$

$[ABBBAB] \to [A][BBBAB]$

$[BBBAB] \to [B][BBAB]$

$[BBAB] \to [B][BAB]$

$[BAB] \to [B][AB]$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

$S \rightarrow [A][ABBBAB]$

$[ABBBAB] \rightarrow [A][BBBAB]$

$[BBBAB] \rightarrow [B][BBAB]$

$[BBAB] \rightarrow [B][BAB]$

$[BAB] \rightarrow [B][AB]$

$[AB] \rightarrow [A][B]$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

$S \rightarrow [A][ABBBAB]$

$[ABBBAB] \rightarrow [A][BBBAB]$

$[BBBAB] \rightarrow [B][BBAB]$

$[BBAB] \rightarrow [B][BAB]$

$[BAB] \rightarrow [B][AB]$

$[AB] \rightarrow [A][B]$

$[A] \rightarrow a$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

$S \to [A][ABBBAB]$

$[ABBBAB] \to [A][BBBAB]$

$[BBBAB] \to [B][BBAB]$

$[BBAB] \to [B][BAB]$

$[BAB] \to [B][AB]$

$[AB] \to [A][B]$

$[A] \to a$

$[B] \to b$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

$S \rightarrow [A][ABBBAB]$

$[ABBBAB] \rightarrow [A][BBBAB]$

$[BBBAB] \rightarrow [B][BBAB]$

$[BBAB] \rightarrow [B][BAB]$

$[BAB] \rightarrow [B][AB]$

$[AB] \rightarrow [A][B]$

$[A] \rightarrow a$

$[B] \rightarrow b$

Think About generalize this to any w of length n.

$\{a, b\}^* a \{a, b\}^n$

1) DFA: exactly 2^{n+1} size DFA. NFA: exactly $n + 2$ states.

$\{a, b\}^* a \{a, b\}^n$

1) DFA: exactly 2^{n+1} size DFA. NFA: exactly $n + 2$ states.

2) CFG: We obtain $O(\log n)$ size.

$\{a, b\}^* a\{a, b\}^n$

1) DFA: exactly 2^{n+1} size DFA. NFA: exactly $n + 2$ states.

2) CFG: We obtain $O(\log n)$ size.

$\{a, b\}^*$ CONCAT $a\{a, b\}^n$

$\{a, b\}^* a\{a, b\}^n$

1) DFA: exactly 2^{n+1} size DFA. NFA: exactly $n + 2$ states.

2) CFG: We obtain $O(\log n)$ size.

$\{a, b\}^*$ CONCAT $a\{a, b\}^n$

$\{a, b\}^* a\{a, b\}^n$

1) DFA: exactly 2^{n+1} size DFA. NFA: exactly $n + 2$ states.

2) CFG: We obtain $O(\log n)$ size.

$\{a, b\}^*$ CONCAT $a\{a, b\}^n$

$\{a, b\}^* a$. Has 5-rule CFG:

$\{a,b\}^*a\{a,b\}^n$

1) DFA: exactly 2^{n+1} size DFA. NFA: exactly $n+2$ states.

2) CFG: We obtain $O(\log n)$ size.

$\{a,b\}^*$ CONCAT $a\{a,b\}^n$

$\{a,b\}^*a$. Has 5-rule CFG:
$a\{a,b\}^n$. A $O(\log n)$ rule CFG.

$\{a,b\}^* a\{a,b\}^n$

1) DFA: exactly 2^{n+1} size DFA. NFA: exactly $n+2$ states.

2) CFG: We obtain $O(\log n)$ size.

$\{a,b\}^*$ CONCAT $a\{a,b\}^n$

$\{a,b\}^* a$. Has 5-rule CFG:
$a\{a,b\}^n$. A $O(\log n)$ rule CFG.
$\{a,b\}^*$ CONCAT $a\{a,b\}^n$ has $O(\log n)$ rule CFG.

Any CFG can be Put Into Chomsky Normal Form

Recall the CFG for $\{a^m b^n : m > n\}$. We put it into Chomsky Normal Form.

1) $S \rightarrow AT$
2) $T \rightarrow aTb$
3) $T \rightarrow e$
4) $A \rightarrow Aa$
5) $A \rightarrow a$

Any CFG can be Put Into Chomsky Normal Form

Recall the CFG for $\{a^m b^n : m > n\}$. We put it into Chomsky Normal Form.

1) $S \rightarrow AT$
2) $T \rightarrow aTb$
3) $T \rightarrow e$
4) $A \rightarrow Aa$
5) $A \rightarrow a$

Use nonterminals $[aT]$, $[b]$, $[a]$. Replace $T \rightarrow aTb$ with:

$T \rightarrow [aT][b]$
$[aT] \rightarrow [a]T$
$[b] \rightarrow b$.
$[a] \rightarrow a$

Any CFG can be Put Into Chomsky Normal Form

Recall the CFG for $\{a^m b^n : m > n\}$. We put it into Chomsky Normal Form.
1) $S \to AT$
2) $T \to aTb$
3) $T \to e$
4) $A \to Aa$
5) $A \to a$
Use nonterminals $[aT]$, $[b]$, $[a]$. Replace $T \to aTb$ with:
$T \to [aT][b]$
$[aT] \to [a]T$
$[b] \to b$.
$[a] \to a$
Repeat the process with the other rules.

Another Measure of Size: Number of NTs

Let $A = \{e, a, a^2, \ldots, a^n\}$

Another Measure of Size: Number of NTs

Let $A = \{e, a, a^2, \ldots, a^n\}$
We did the following:

Another Measure of Size: Number of NTs

Let $A = \{e, a, a^2, \ldots, a^n\}$
We did the following:

1) $\exists$ a CFG G, $L(G) = A$, G has $O(n^{1/3})$ Non Terminals.

Another Measure of Size: Number of NTs

Let $A = \{e, a, a^2, \ldots, a^n\}$
We did the following:

1) $\exists$ a CFG G, $L(G) = A$, G has $O(n^{1/3})$ Non Terminals.

2) We can use this CFG to obtain the following:
$\forall X \subseteq \{e, a, \ldots, a^n\}$ $\exists$ a CFG for X with $O(n^{1/3})$ Non Terminals.

Another Measure of Size: Number of NTs

Let $A = \{e, a, a^2, \ldots, a^n\}$
We did the following:

1) $\exists$ a CFG G, $L(G) = A$, G has $O(n^{1/3})$ Non Terminals.

2) We can use this CFG to obtain the following:
$\forall X \subseteq \{e, a, \ldots, a^n\}$ $\exists$ a CFG for X with $O(n^{1/3})$ Non Terminals.

3) $\exists X \subseteq \{e, a, \ldots, a^n\}$ $\forall$ CFG's for X have $\Omega(n^{1/3})$ Non Terminals.

Another Measure of Size: Number of NTs

Let $A = \{e, a, a^2, \ldots, a^n\}$
We did the following:

1) $\exists$ a CFG G, $L(G) = A$, G has $O(n^{1/3})$ Non Terminals.

2) We can use this CFG to obtain the following:
$\forall X \subseteq \{e, a, \ldots, a^n\}$ $\exists$ a CFG for X with $O(n^{1/3})$ Non Terminals.

3) $\exists X \subseteq \{e, a, \ldots, a^n\}$ $\forall$ CFG's for X have $\Omega(n^{1/3})$ Non Terminals.

4) The CFG for A was useful but not optimal. In HW 7 you got a CFG for A that has substantially less nonterminals then $O(n^{1/3})$. Will go over that Thursday.

MISC

1) If L_1 is a CFL and L_2 is regular then $L_1 \cap L_2$ is a CFL.

MISC

1) If L_1 is a CFL and L_2 is regular then $L_1 \cap L_2$ is a CFL.

2) Recall: DFA's are **Recognizers**, Regex are **Generators**.
CFG's are **Generators**. There is a **Recognizer** equivalent to it:

PDA: Push Down Automata

.

They are NFAs with a stack.

MISC

1) If L_1 is a CFL and L_2 is regular then $L_1 \cap L_2$ is a CFL.

2) Recall: DFA's are **Recognizers**, Regex are **Generators**.
CFG's are **Generators**. There is a **Recognizer** equivalent to it:

PDA: Push Down Automata

.

They are NFAs with a stack.
The proof that PDA-recognizers and CFG-generators are equivalent is messy so we won't be doing it. We won't deal with PDA's in this course at all.

CNF for $\{w\}$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

$S \rightarrow [A][ABBBAB]$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

$S \rightarrow [A][ABBBAB]$

$[ABBBAB] \rightarrow [A][BBBAB]$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

$S \rightarrow [A][ABBBAB]$

$[ABBBAB] \rightarrow [A][BBBAB]$

$[BBBAB] \rightarrow [B][BBAB]$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

$S \to [A][ABBBAB]$

$[ABBBAB] \to [A][BBBAB]$

$[BBBAB] \to [B][BBAB]$

$[BBAB] \to [B][BAB]$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

$S \to [A][ABBBAB]$

$[ABBBAB] \to [A][BBBAB]$

$[BBBAB] \to [B][BBAB]$

$[BBAB] \to [B][BAB]$

$[BAB] \to [B][AB]$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

$S \rightarrow [A][ABBBAB]$

$[ABBBAB] \rightarrow [A][BBBAB]$

$[BBBAB] \rightarrow [B][BBAB]$

$[BBAB] \rightarrow [B][BAB]$

$[BAB] \rightarrow [B][AB]$

$[AB] \rightarrow [A][B]$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

$S \rightarrow [A][ABBBAB]$

$[ABBBAB] \rightarrow [A][BBBAB]$

$[BBBAB] \rightarrow [B][BBAB]$

$[BBAB] \rightarrow [B][BAB]$

$[BAB] \rightarrow [B][AB]$

$[AB] \rightarrow [A][B]$

$[A] \rightarrow a$

CNF for $\{aabbbab\}$

Example CNF for $\{aabbbab\}$

$S \to [A][ABBBAB]$

$[ABBBAB] \to [A][BBBAB]$

$[BBBAB] \to [B][BBAB]$

$[BBAB] \to [B][BAB]$

$[BAB] \to [B][AB]$

$[AB] \to [A][B]$

$[A] \to a$

$[B] \to b$

CNF for $\{w\}$

CNF for $\{w\}$

1. You can do something similar for any w.

CNF for $\{w\}$

1. You can do something similar for any w.
2. If $|w| = n$ then the CFG will be $O(n)$ rules.

CNF for $\{w\}$

1. You can do something similar for any w.
2. If $|w| = n$ then the CFG will be $O(n)$ rules.
3. Question we will come back to LATER:
   $(\exists w)$ such that $\{w\}$ requires large CFG?

CFL $\subset$ P

Poly Time Algorithm for CFG Membership

Let L be a CFL. Let G be the Chomsky Normal Form CFG for L.

Poly Time Algorithm for CFG Membership

Let L be a CFL. Let G be the Chomsky Normal Form CFG for L.

$w = \sigma_1 \cdots \sigma_n$.

Poly Time Algorithm for CFG Membership

Let L be a CFL. Let G be the Chomsky Normal Form CFG for L.

$w = \sigma_1 \cdots \sigma_n$.
We want to know if $w \in L$. We assume $w \neq e$.

Poly Time Algorithm for CFG Membership

Let L be a CFL. Let G be the Chomsky Normal Form CFG for L.

$w = \sigma_1 \cdots \sigma_n$.
We want to know if $w \in L$. We assume $w \neq e$.

For $i \leq j$ let

$$\mathrm{GEN}[i,j] = \{A : A \Rightarrow \sigma_i \cdots \sigma_j\}$$

Poly Time Algorithm for CFG Membership

Let L be a CFL. Let G be the Chomsky Normal Form CFG for L.

$w = \sigma_1 \cdots \sigma_n$.
We want to know if $w \in L$. We assume $w \neq e$.

For $i \le j$ let

$$\mathrm{GEN}[i,j] = \{A : A \Rightarrow \sigma_i \cdots \sigma_j\}$$

We will find **all** $\mathrm{GEN}[i,j]$. Hence we will find $\mathrm{GEN}[1,n]$. Hence we will find if $S \in \mathrm{GEN}[1,n]$.

Bottom Up View

Bottom Up View

$$\sigma_1 \cdots \sigma_{i-1} \overbrace{\sigma_i}^{A} \sigma_{i+1} \cdots \sigma_n$$

Bottom Up View

$$\sigma_1 \cdots \sigma_{i-1} \overbrace{\sigma_i}^{A} \sigma_{i+1} \cdots \sigma_n$$

$$\mathrm{GEN}[i,i] \; = \; \{A : A \rightarrow \sigma_i\}$$

Bottom Up View

$$\sigma_1 \cdots \sigma_{i-1} \overbrace{\sigma_i}^{A} \sigma_{i+1} \cdots \sigma_n$$

$$\mathrm{GEN}[i, i] \;=\; \{A : A \rightarrow \sigma_i\}$$

$$\sigma_1 \cdots \sigma_{i-1} \overbrace{\sigma_i}^{B} \overbrace{\sigma_{i+1}}^{C} \sigma_{i+2} \cdots \sigma_n$$

Bottom Up View

$$\sigma_1 \cdots \sigma_{i-1} \overbrace{\sigma_i}^{A} \sigma_{i+1} \cdots \sigma_n$$

$$\mathrm{GEN}[i,i] = \{A : A \to \sigma_i\}$$

$$\sigma_1 \cdots \sigma_{i-1} \overbrace{\sigma_i}^{B} \overbrace{\sigma_{i+1}}^{C} \sigma_{i+2} \cdots \sigma_n$$

$$\mathrm{GEN}[i,i+1] = \{A : A \to BC \ \wedge \ B \to \sigma_i \ \wedge \ C \to \sigma_{i+1}\}$$

Bottom Up View

$$\sigma_1 \cdots \sigma_{i-1} \overbrace{\sigma_i}^{A} \sigma_{i+1} \cdots \sigma_n$$

$$\mathrm{GEN}[i,i] = \{A : A \to \sigma_i\}$$

$$\sigma_1 \cdots \sigma_{i-1} \overbrace{\sigma_i}^{B} \overbrace{\sigma_{i+1}}^{C} \sigma_{i+2} \cdots \sigma_n$$

$$\begin{aligned} \mathrm{GEN}[i,i+1] &= \{A : A \to BC \ \wedge \ B \to \sigma_i \ \wedge \ C \to \sigma_{i+1}\} \\ &= \{A : A \to BC \\ &\qquad \wedge \ B \in GEN[i,i] \ \wedge \ C \in GEN[i+1,i+1]\} \end{aligned}$$

Recurrence

Recurrence

$$\mathrm{GEN}[i,j] = \{A : A \Rightarrow \sigma_i \cdots \sigma_j\}$$

Recurrence

$$\mathrm{GEN}[i,j] = \{A : A \Rightarrow \sigma_i \cdots \sigma_j\}$$

$$\sigma_1 \cdots \sigma_{i-1} \overbrace{\sigma_i \sigma_{i+1} \cdots \sigma_k}^{B} \overbrace{\sigma_{k+1} \sigma_{k+2} \cdots \sigma_j}^{C} \sigma_{j+1} \cdots \sigma_n$$

Recurrence

$$\mathrm{GEN}[i,j] = \{A : A \Rightarrow \sigma_i \cdots \sigma_j\}$$

$$\sigma_1 \cdots \sigma_{i-1} \overbrace{\sigma_i \sigma_{i+1} \cdots \sigma_k}^{B} \overbrace{\sigma_{k+1} \sigma_{k+2} \cdots \sigma_j}^{C} \sigma_{j+1} \cdots \sigma_n$$

$$\mathrm{GEN}[i,j] \quad = \quad \bigcup_{i \le k < j} \{A : A \to BC \ \wedge \ B \Rightarrow \sigma_i \cdots \sigma_k \ \wedge \ C \Rightarrow \sigma_{k+1} \cdots \sigma_j\}$$

Recurrence

$$\mathrm{GEN}[i,j] = \{A : A \Rightarrow \sigma_i \cdots \sigma_j\}$$

$$\sigma_1 \cdots \sigma_{i-1} \overbrace{\sigma_i \sigma_{i+1} \cdots \sigma_k}^{B} \overbrace{\sigma_{k+1} \sigma_{k+2} \cdots \sigma_j}^{C} \sigma_{j+1} \cdots \sigma_n$$

$$\begin{aligned} \mathrm{GEN}[i,j] &= \bigcup_{i \le k < j} \{A : A \to BC \ \wedge \ B \Rightarrow \sigma_i \cdots \sigma_k \ \wedge \ C \Rightarrow \sigma_{k+1} \cdots \sigma_j\} \\ &= \bigcup_{i \le k < j} \{A : A \to BC \ \wedge \ B \in \mathrm{GEN}[i,k] \ \wedge \ C \in \mathrm{GEN}[k+1,j]\} \end{aligned}$$

The Algorithm

```
for i = 1 to n do
    for j = i to n do
        GEN[i,j] ← ∅

for i = 1 to n do
    for all rules A → σ_i do
        GEN[i,i] ← GEN[i,i] with A

for s = 2 to n do
    for i = 1 to n-s+1 do
        j ← i+s-1 do
        for k = i to j-1 do
            for all rules A → BC
                where B ∈ GEN[i,k] and C ∈ GEN[k+1,j]
                    GEN[i,j] ← GEN[i,j] with A
```