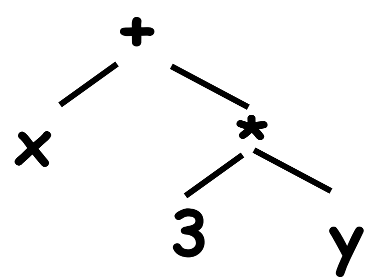
言語処理系論

小林　直樹

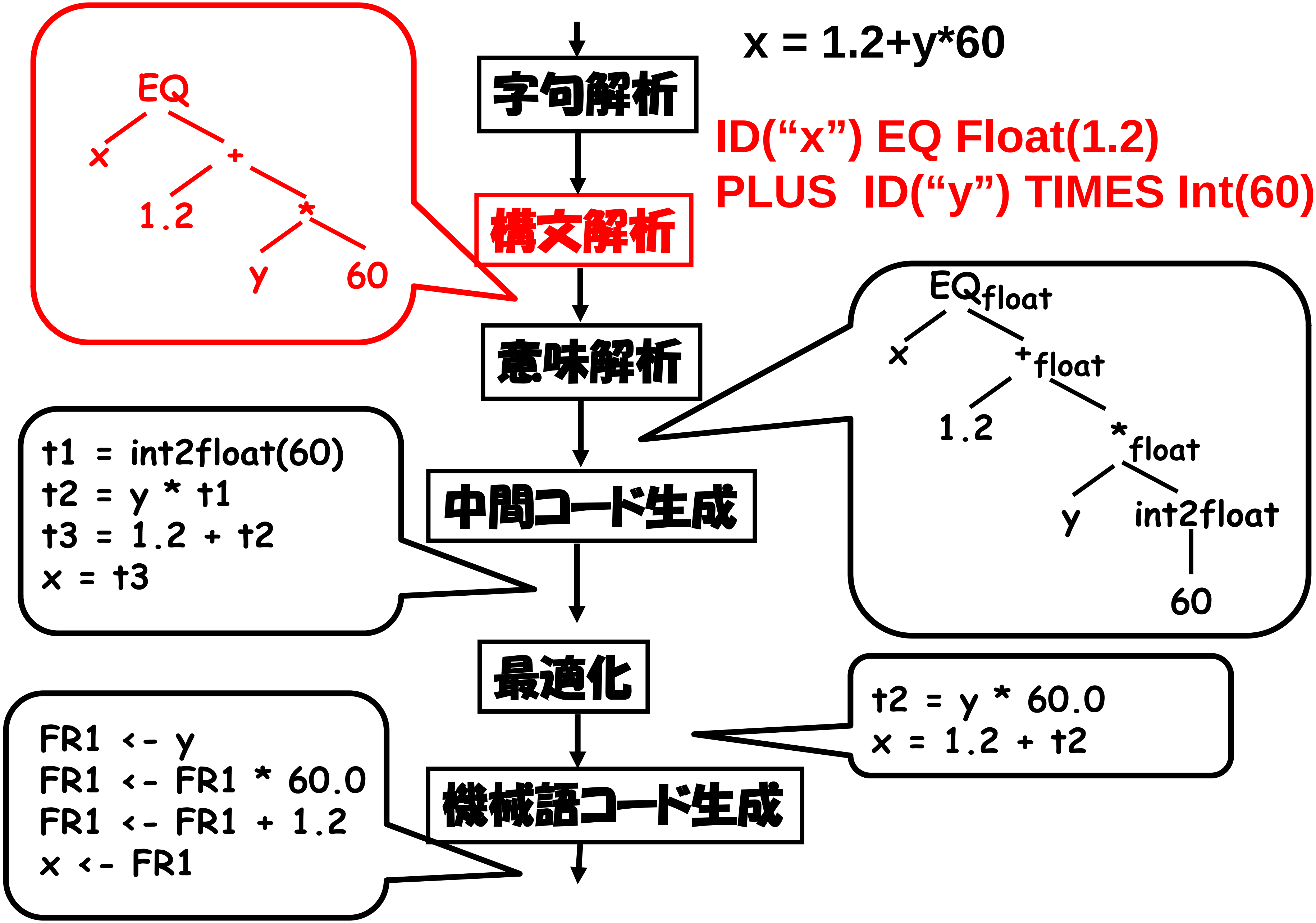
本日の内容

- 構文解析
  - ・トークンの列を論理的構造を表す木（AST、抽象構文木）に変換

ID(“x”) + Int(3) * ID(“y”) ⟶



コンパイラの構造



x = 1.2+y*60
字句解析
ID("x") EQ Float(1.2)
PLUS ID("y") TIMES Int(60)
構文解析
EQ
x
+
1.2
*
y
60
意味解析
EQfloat
x
+float
1.2
*float
y
int2float
60
中間コード生成
t1 = int2float(60)
t2 = y * t1
t3 = 1.2 + t2
x = t3
最適化
t2 = y * 60.0
x = 1.2 + t2
機械語コード生成
FR1 <- y
FR1 <- FR1 * 60.0
FR1 <- FR1 + 1.2
x <- FR1

アウトライン

- 基本原則と例外
  （および文脈自由文法の復習）
- 文法の曖昧性
- 構文解析アルゴリズム
  - CKY法
  - トップダウン構文解析 LL(k)

構文解析の基本原則

- 文脈自由文法を用いて構文を定義

  E → id | num | E+E | E * E | (E)

- 与えられた入力トークン列が文脈脈自由文法によって生成できるかを判定。
  生成できればその導出木を出力。

文脈自由文法（復習）

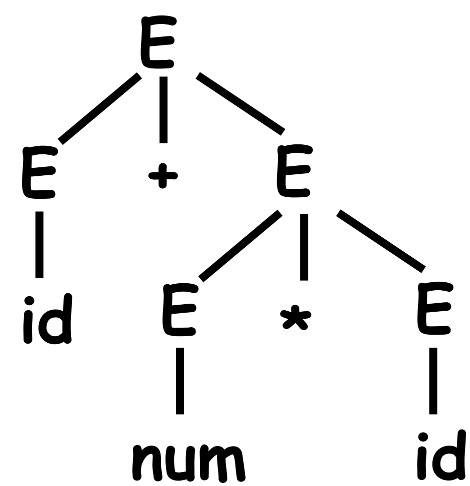
- ４つ組 $G = (\Sigma, N, R, S)$
  - Σ: 終端記号（ここではトークン）の集合
  - N: 非終端記号の集合
  - R: 生成規則$A \to \alpha$ ($A \in N$, $\alpha \in (\Sigma \cup N)^*$)の集合
  - S: 開始記号
- Gが生成する言語：
  生成規則を使ってSを書き換えて得られる語の集合
  $L(G) = \{w \in \Sigma^* \mid S \to^* w\}$

導出列と構文解析木

- 導出列：開始記号から語 $w \in \Sigma^*$ への書き換え列

  E → E+E → E+E*E → id+E*E→ id+num*E → id+num*id

- 構文解析木（具象構文木）
  - 導出の過程を木構造で表現したもの



  - 意味のない書き換え順序による違いを除去
    - E → E+E → E+E*E → id+E*E→ id+num*E → id+num*id
    - E → E+E → id+E → id+E*E→ id+num*E → id+num*id

構文解析の基本原則

- 文脈自由文法を用いて構文を定義

  E → id | num | E+E | E * E | (E)

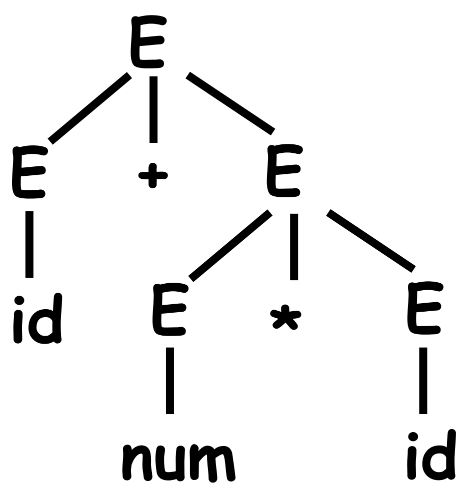
- 与えられた入力トークン列が文脈脈自由文法によって生成できるかを判定。
  生成できればその導出木を出力。

基本原則からはずれる点(1)

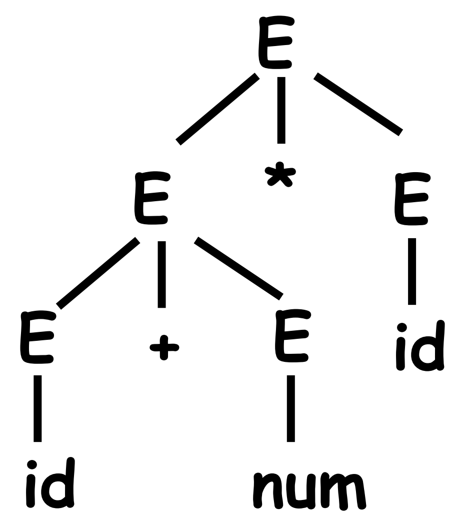
- 曖昧性への対応

  $E \rightarrow \mathrm{id} \mid \mathrm{num} \mid E+E \mid E * E \mid (E)$

  id+num*id は



か



か？

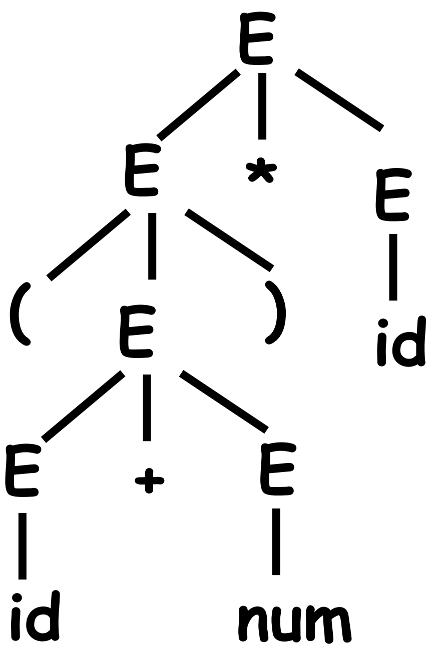
対処法（後述）：

- 曖昧性のない文法で記述
- 文法＋演算子の優先順位の宣言

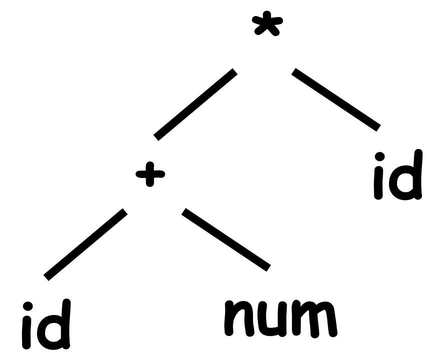
基本原則からはずれる点(2)

- 出力は具象構文木（構文解析木）ではなく、余計な情報を取り除いた**抽象構文木(AST)**

入力：(id + num) * id



具象構文木



抽象構文木

対処：
具象構文木をたどりながら
抽象構文木を構成

基本原則からはずれる点(3)

- プログラミング言語の構文記述には、
  文脈自由文法のサブクラスで十分
- 構文解析の効率が重要
  （プログラミング言語の場合、
  　入力に対してほぼ線形時間であってほしい）

-> 種々の構文解析アルゴリズム

(CKY, LL(k), LR(k), LALR, ...)

アウトライン

- 基本原則と例外
  （および文脈自由文法の復習）
- 文法の曖昧性
- 構文解析アルゴリズム
  - CKY法
  - トップダウン構文解析 LL(k)

曖昧な文法の例

$E \rightarrow$ id | num | E-E | E * E | (E)

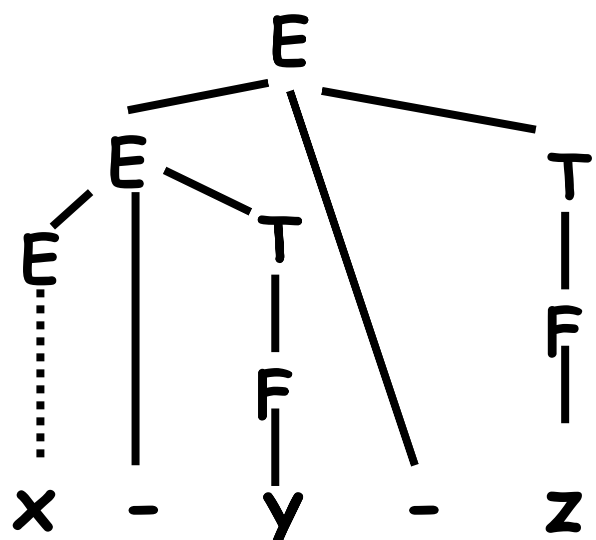
- x-y-z は(x-y)-z か x-(y-z)か？
- x-y*z は(x-y)*z か x-(y*z)か？

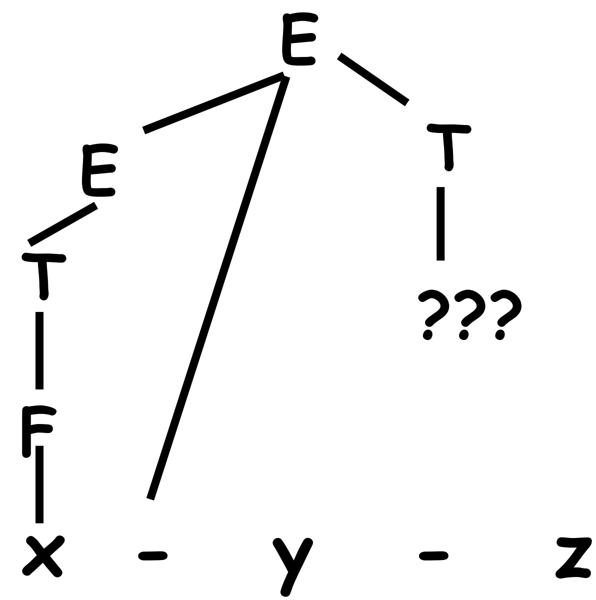
曖昧な文法に対する対処

$E \to id \mid num \mid E-E \mid E * E \mid (E)$

- 曖昧でない文法に（手で）書き換え

$E \to T \mid E-T$	任意の式
$T \to F \mid T*F$	一番外側に"-"は不可
$F \to id \mid num \mid (E)$	一番外側に２項演算子は不可





曖昧な文法に対する対処

$E \rightarrow id \mid num \mid E\text{-}E \mid E * E \mid (E)$

- 曖昧でない文法に（手で）書き換え

$E \rightarrow T \mid E\text{-}T$	任意の式
$T \rightarrow F \mid T*F$	一番外側に"-"は不可
$F \rightarrow id \mid num \mid (E)$	一番外側に２項演算子は不可

- 演算子の優先度や結合法則を指定
（曖昧でない文法に自動変換または優先度を考慮できる構文解析アルゴリズムを使用）
  - 優先順位：＊＞－
  - 結合法則：＊，－は左結合

練習問題

・次の文法を曖昧性がないものに直せ。

$F \to \mathrm{id} \mid \neg F \mid F \wedge F \mid F \Rightarrow F \mid (F)$

ただし演算子の優先度は　$\neg > \wedge > \Rightarrow$

$\wedge$は左結合、$\Rightarrow$は右結合

練習問題（解答例）

- 次の文法を曖昧性がないものに直せ。

  $F \rightarrow id \mid \neg F \mid F \wedge F \mid F \Rightarrow F$

  ただし演算子の優先度は　$\neg > \wedge > \Rightarrow$

  $\wedge$は左結合、 $\Rightarrow$は右結合

$F \rightarrow G \mid G \Rightarrow F$

$G \rightarrow H \mid G \wedge H$

$H \rightarrow id \mid \neg H \mid (F)$

演算子の優先度の反映

$E \rightarrow E\ op_1\ E |\ E\ op_2\ E | \ldots |\ E\ op_n\ E |\ id\ |\ (E)$

優先度：$op_n > \ldots > op_1$

非終端記号$E=E_1, E_2, \ldots, E_{n+1}$**を用意**

E_k：**一番外側に現れてよいのは優先度が**k**以上のもののみ**

$E_k \rightarrow E_{k+1}\ \ |\ \ E_k\ op_k\ E_{k+1}$　　　op_k**が左結合の場合**

$E_k \rightarrow E_{k+1}\ \ |\ \ E_{k+1}\ op_k\ E_k$　　　op_k**が右結合の場合**

$E_{n+1} \rightarrow id\ |\ (E_1)$

本質的に曖昧な文脈自由言語

- 文脈自由言語によっては曖昧性は回避不能

例：$L = \{a^m b^k c^n d^k \mid k,m,n \geq 1\}$
$\cup \{a^k b^m c^k d^n \mid k,m,n \geq 1\}$

曖昧な文脈自由文法では記述可能

$S \to A\ E \mid F\ D$

$A \to a \mid aA$ $\quad$ $D \to d \mid dD$

$E \to C \mid bEd$ $\quad$ $C \to c \mid cC$

$F \to B \mid aFc$ $\quad$ $B \to b \mid bB$

曖昧でない文脈自由文法では記述不能

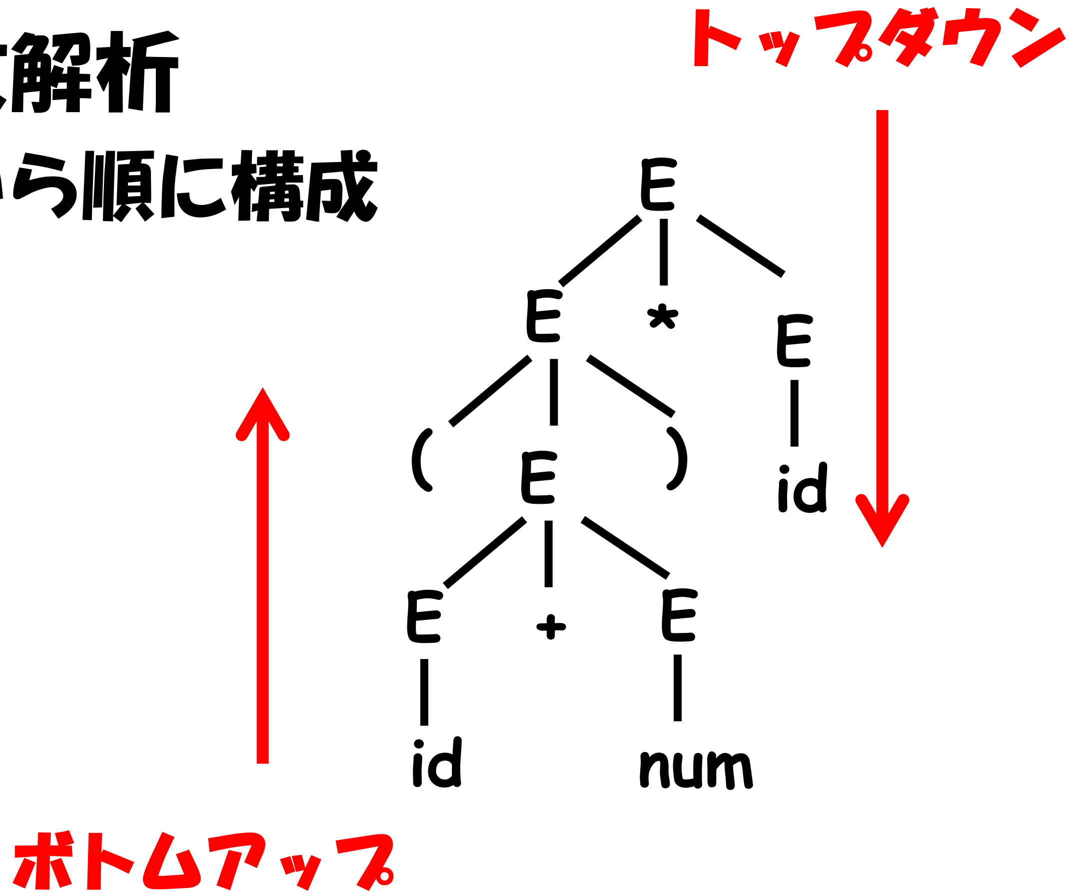
(cf. R.J.Parikh, “On Context-Free Languages”, JACM, 1966)

アウトライン

- 基本原則と例外
  （および文脈自由文法の復習）
- 文法の曖昧性
- 構文解析アルゴリズム
  - CKY法
  - トップダウン構文解析 LL(k)

トップダウンvsボトムアップ解析

- トップダウン構文解析
  - 構文解析木を上から順に構成
- ボトムアップ構文解析
  - 構文解析木を下から順に構成



CKY法
(Cocke-Kasami-Younger)

- 一般の文脈自由言語を扱えるので自然言語など曖昧性がある複雑な文法の解析に使用
- ボトムアップ構文解析の一種
  - 入力の各部分列に対する解析結果を表(CKY表)で保持し、それを組み合わせて長い部分列の構造を計算
- 計算コストは入力サイズnに対して$O(n^3)$

例

文法：$E \rightarrow T \mid E+T \quad T \rightarrow F \mid T*F \quad F \rightarrow id \mid (E)$

入力：$\overbrace{id}^{F\ \ T\ \ E}$ + id * id

例

文法：$E \to T \mid E+T \quad T \to F \mid T*F \quad F \to id \mid (E)$

入力：$\overbrace{id}^{F\ T\ E} \quad + \quad \overbrace{id}^{F\ T\ E} \quad * \quad id$

例

文法：$E \rightarrow T \mid E+T \quad T \rightarrow F \mid T*F \quad F \rightarrow id \mid (E)$

入力：$\overbrace{id}^{F\ T\ E} \quad + \quad \overbrace{id}^{F\ T\ E} \quad * \quad \overbrace{id}^{F\ T\ E}$

例

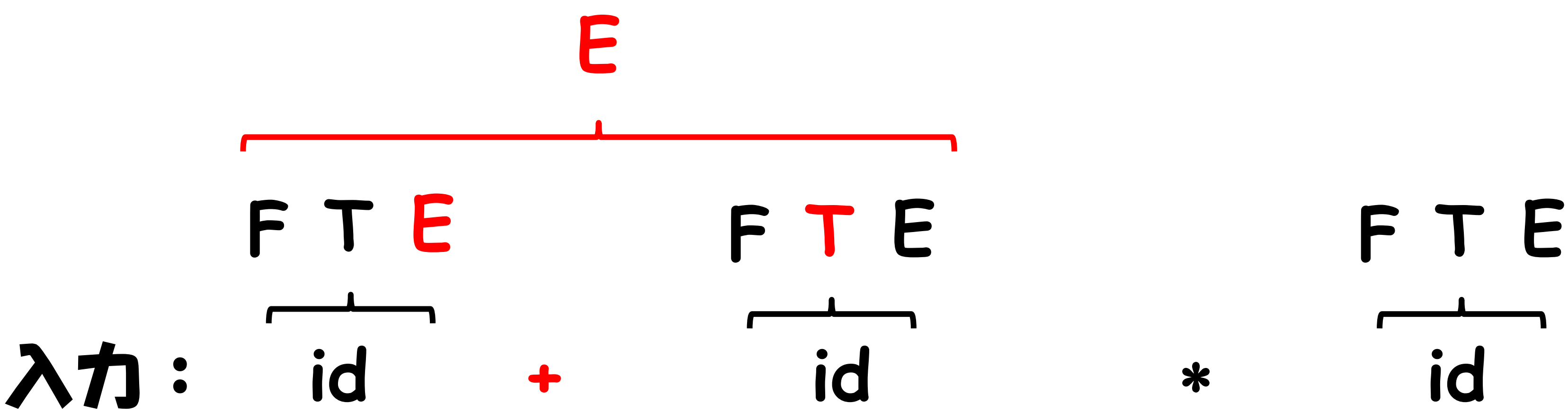
文法：$E \rightarrow T \mid E+T \quad T \rightarrow F \mid T*F \quad F \rightarrow id \mid (E)$

F T E　　F T E　　F T E

入力：id　+　id　*　id

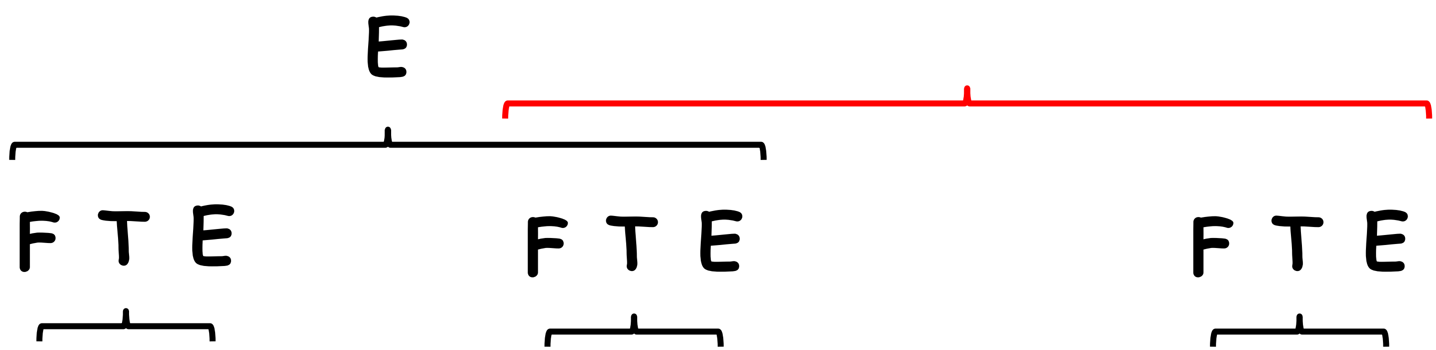
例

文法：$E \rightarrow T \mid E+T \quad T \rightarrow F \mid T*F \quad F \rightarrow id \mid (E)$



例

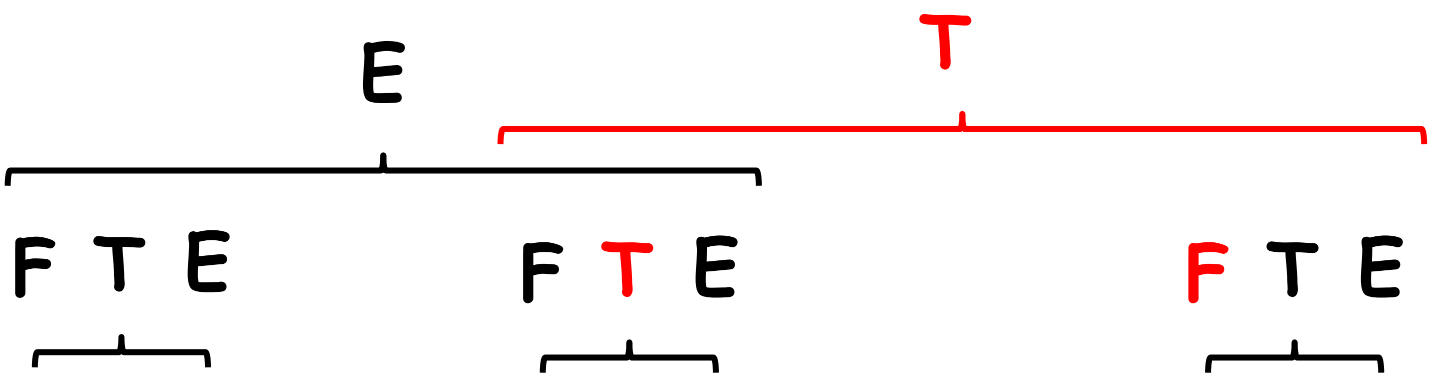
文法：$E \rightarrow T \mid E+T \quad T \rightarrow F \mid T*F \quad F \rightarrow id \mid (E)$



入力：id + id * id

例

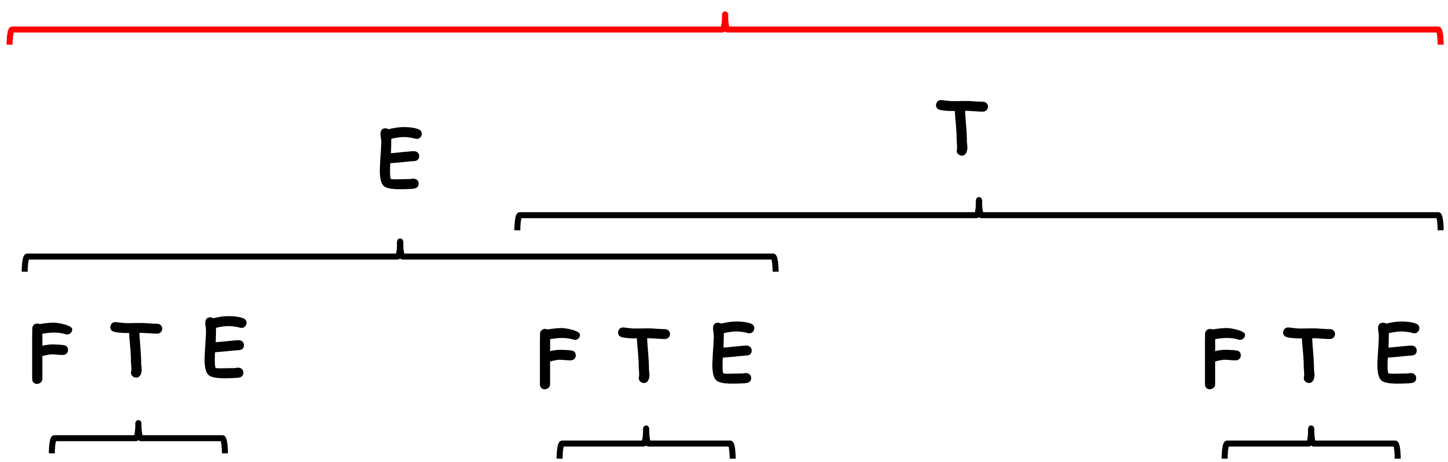
文法：$E \rightarrow T \mid E+T \quad T \rightarrow F \mid T*F \quad F \rightarrow id \mid (E)$



入力：id + id * id

例

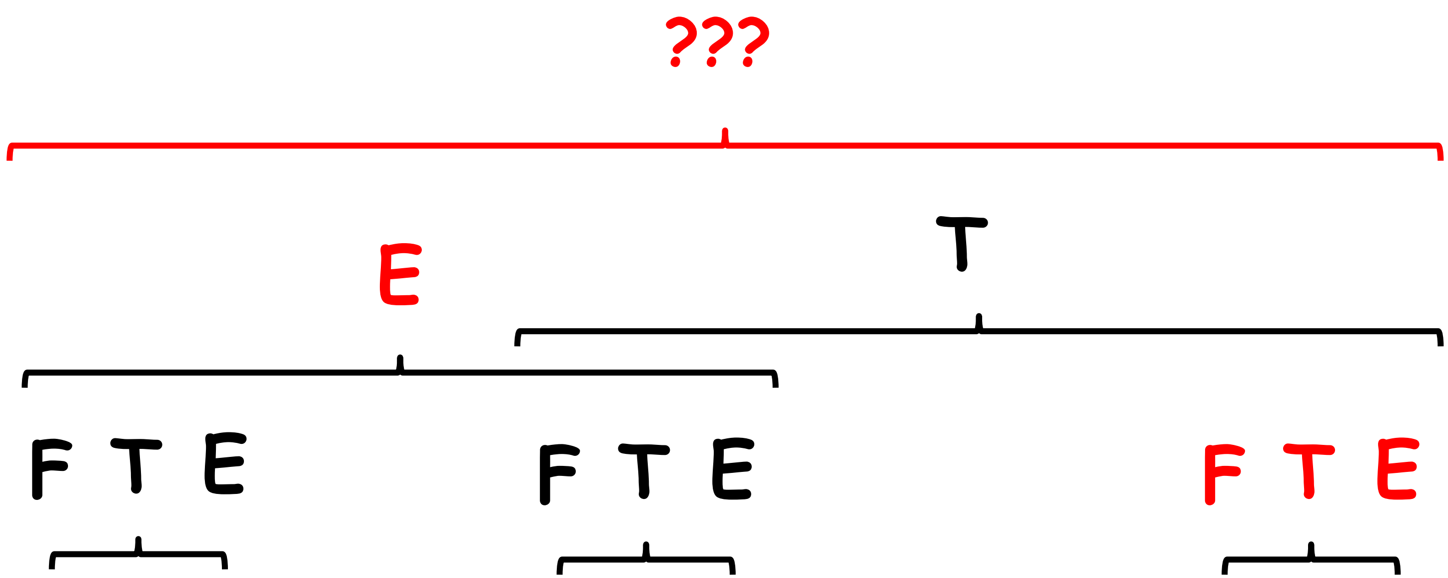
文法： $E \rightarrow T \mid E{+}T \quad T \rightarrow F \mid T{*}F \quad F \rightarrow id \mid (E)$



入力： id + id * id

例

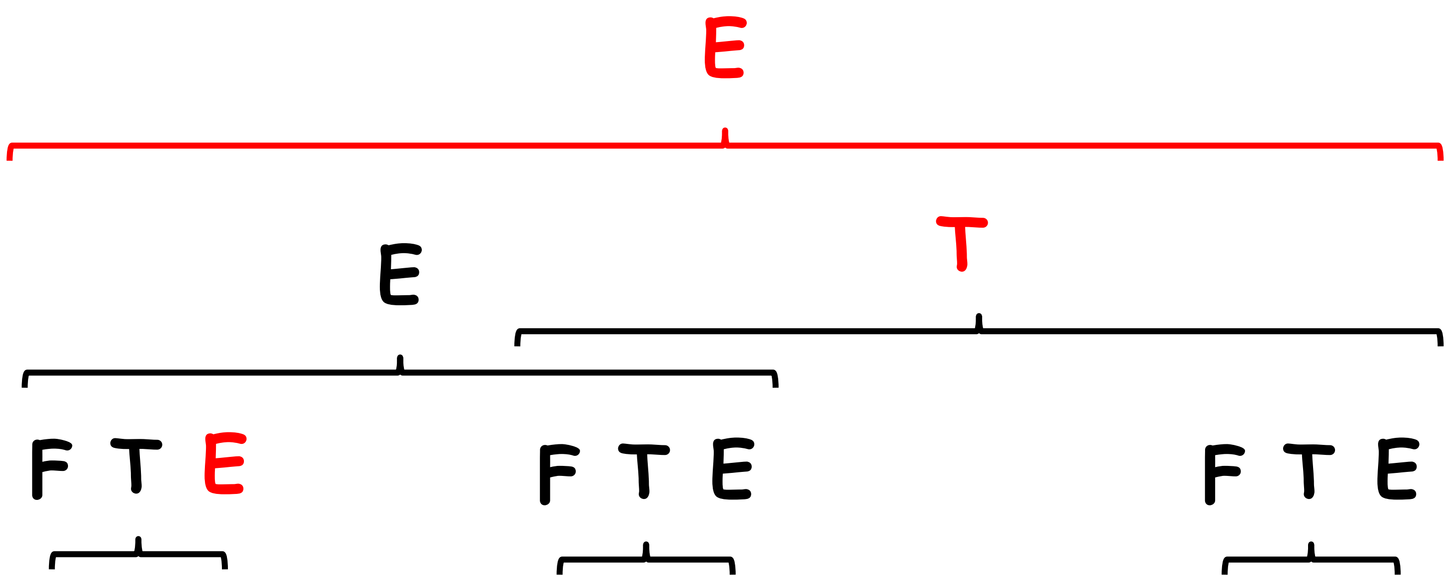
文法：$E \to T \mid E+T \quad T \to F \mid T*F \quad F \to id \mid (E)$



入力：id + id * id

例

文法：$E \to T \mid E{+}T \quad T \to F \mid T{*}F \quad F \to id \mid (E)$



入力： id + id * id

例2

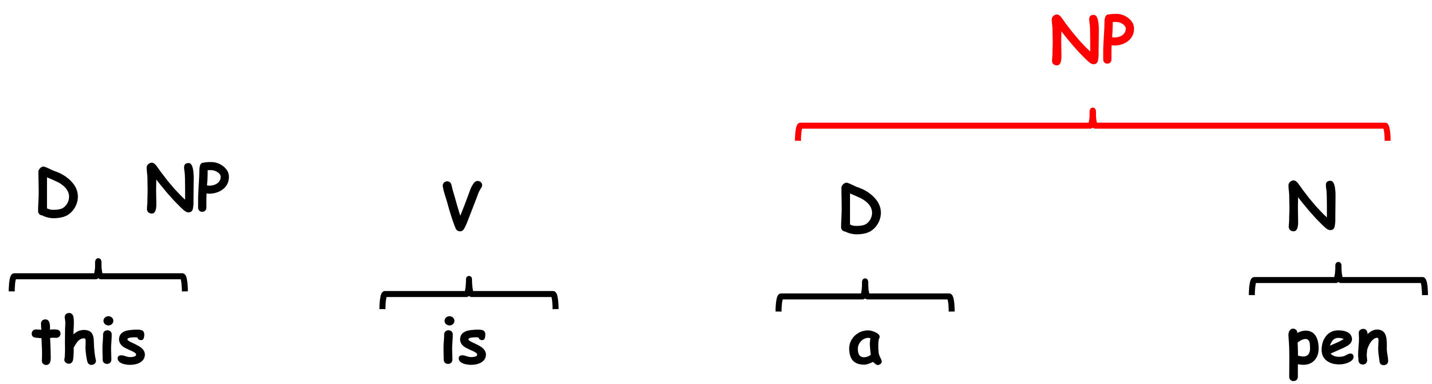
$S \rightarrow NP\ VP \quad NP \rightarrow D\ N \mid this \quad VP \rightarrow V\ NP \mid V\ A$

$V \rightarrow is \quad D \rightarrow a \mid this \quad N \rightarrow pen \quad A \rightarrow black$

例2

S → NP VP   NP → D N | this   VP → V NP | V A

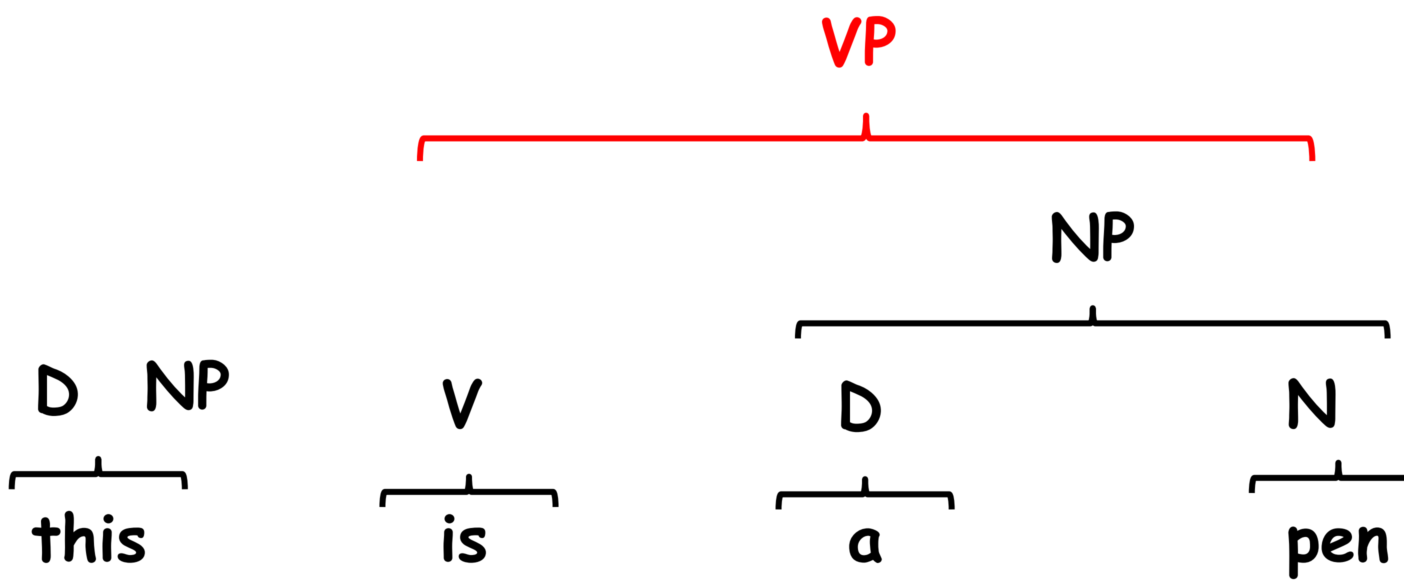
V → is   D → a | this   N → pen   A → black



例2

$S \rightarrow NP\ VP \quad NP \rightarrow D\ N \mid this \quad VP \rightarrow V\ NP \mid V\ A$

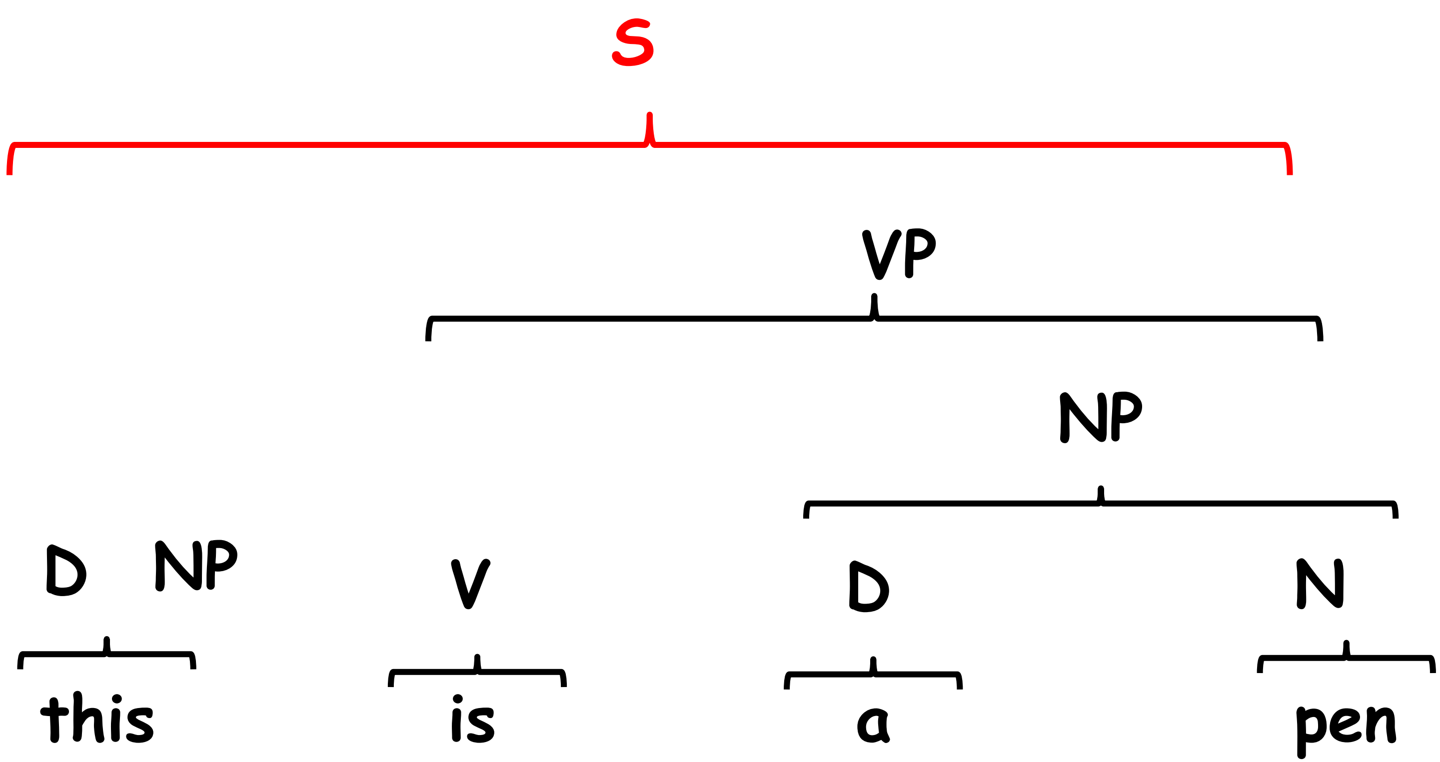
$V \rightarrow is \quad D \rightarrow a \mid this \quad N \rightarrow pen \quad A \rightarrow black$



例2

S → NP VP   NP → D N | this   VP → V NP | V A

V → is   D → a | this   N → pen   A → black



CKY表

- (i,j)の要素にiからj番目の文字（トークン）に対する解析結果を記入

	1:this	2:is	3:a	4:pen
1	D NP			
2		V		
3			D	
4				N

S → NP VP
NP → D N | this
VP → V NP | V A
V → is
D → a | this
N → pen

CKY表

- (i,j)の要素にiからj番目の文字（トークン）に対する解析結果を記入

	1:this	2:is	3:a	4:pen
1	D NP			
2		V		
3			D	NP
4				N

S → NP VP
NP → D N | this
VP → V NP | V A
V → is
D → a | this
N → pen

CKY表

- (i,j)の要素にiからj番目の文字（トークン）に対する解析結果を記入

	1:this	2:is	3:a	4:pen
1	D NP			
2		V		VP
3			D	NP
4				N

S → NP VP
NP → D N | this
VP → V NP | V A
V → is
D → a | this
N → pen

CKY表

- (i,j)の要素にiからj番目の文字（トークン）に対する解析結果を記入

	1:this	2:is	3:a	4:pen
1	D NP			S
2		V		VP
3			D	NP
4				N

S → NP VP
NP → D N | this
VP → V NP | V A
V → is
D → a | this
N → pen

注：構文解析木を出力したい場合には
終端記号のみでなく、解析木をエントリに記入

CKYアルゴリズム（文法はチョムスキー標準形を仮定）

入力 I[1..n]　CKY**表** T[i,j]

```
T[i,j] := {} for each i,j
for each i in {1,…,n} do   /* 長さ1の部分列に対する処理 */
   T[i,i] := {N | N → I[i] ∈R};
for len=1 to n-1 do /* 長さlen+1の部分列に対する処理 */
  for i=1 to n-len do      /* T[i,i+len]の計算 */
    for k=i to n-len do  /* I[i..k], I[k+1..i+len]の結果を統合 */
      for each A → BC ∈R do
        if B∈T[i,k] & C ∈T[k+1,i+len] then
          T[i,i+len] := T[i,i+len] ∪ {A}
```

CKYアルゴリズム（文法はチョムスキー標準形を仮定）

入力 I[1..n]　CKY**表** T[i,j]

T[i,j] := {} for each i,j

for each i in {1,…,n} do　/* **長さ1の部分列に対する処理** */

T[i,i] := {N | N → I[i] ∈R};

for len=1 to n-1 do /* 長

for i=1 to n-len do

for k=i to n-len do　を統合 */

for each A → BC ∈R do

if B∈T[i,k] & C ∈T[k+1,i+len] then

T[i,i+len] := T[i,i+len] ∪ {A}

3重ループなので、
計算コストはO(n^3)
（文法のサイズは固定）

CKY法の利点と欠点

- 利点
  - 任意の文脈自由文法を扱える
- 欠点
  - （コンパイラに用いるには）効率が悪い
    ($O(n^3)$のアルゴリズムは、数万行のソースプログラムの処理には不向き）

練習問題

"this pen is black" に対するCKY表を書け。
ただし文法は以下のとおりとする

S -> NP VP
NP -> D N
VP -> V NP
V -> is
D -> a
N -> pen
A -> black

NP -> this
VP -> V A
D -> this

アウトライン

- 基本原則と例外
  （および文脈自由文法の復習）
- 文法の曖昧性
- 構文解析アルゴリズム
  - CKY法
  - トップダウン構文解析
    - ナイーブなアルゴリズム（バックトラックあり）
    - 先読みつき（LL(k)）

トップダウン構文解析

- 入力を左（先頭）から走査しながらトップダウンに構文解析木を構築

例：E → T | T+E　T → F | F*T　　F → id | (E)

各非終端記号ごとに
それを認識する関数を用意

文法規則に従って再帰呼び出し

```
parseE() =
   let p1=parseT() in
   match next() with '+' -> Plus(p1,parseE()) | _ -> p1
parseT() =
   let p1=parseF() in
   match next() with '*' -> Times(p1,parseT()) | _ -> p1
parseF() =
   match next() with id->Id
                   | '('-> let p=parseE() in next();p
```

ナイーブなトップダウン解析の利点と欠点

- 利点：わかりやすい
- 欠点1：一般にはバックトラックが必要

  E → T | G　　G → T | E　　...

```
parseE() = try parseT()
           with ERROR -> (* 失敗したら *)
            backtrack(); parseG()
```

- 欠点2：左再帰があるとうまくいかない

  E → E+T | ...

```
parseE() = parseE(); ...            (* 無限再帰 *)
```

トップダウン解析の欠点と対処

- 一般にはバックトラックが必要（効率が悪い）

  E → T | G　　G → T | E　　...

```
parseE() = try parseT()
           with ERROR -> (* 失敗したら *)
            backtrack(); parseG()
```

先読みを行ってバックトラックを避ける
（先読みで決定できない文法は扱わない） => LL(k) （後述）

- 左再帰があるとうまくいかない

  E → E+T | ...

```
parseE() = parseE(); ...                (* 無限再帰 *)
```

左再帰の除去（cf. グライバッハの標準形）

左再帰の除去

- **元の文法例:**

  $E \to T \mid E{+}T \quad T \to F \mid T{*}F \qquad F \to id \mid (E)$

- **左再帰除去後:**

  $E \to T \mid T\,G \qquad G \to +T \mid +T\,G$ (* "+T"の有限列 *)

  $T \to F \mid F\,H \qquad H \to *F \mid *F\,H$ (* "*F"の有限列 *)

  $F \to id \mid (E)$

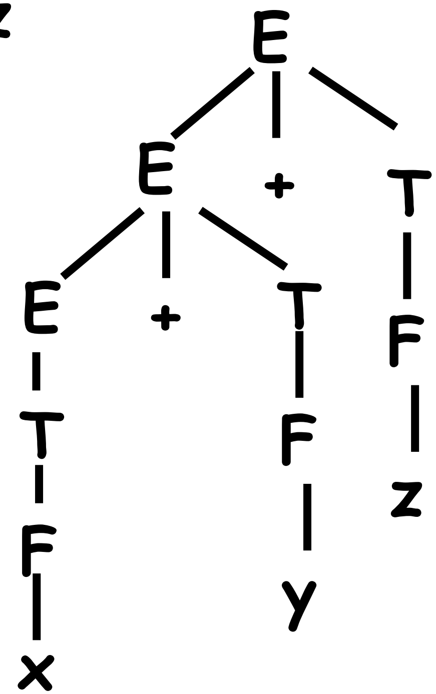
- **一般の場合:**

  グライバッハの標準形を使えば必ず除去可能
  （時間があれば後述）

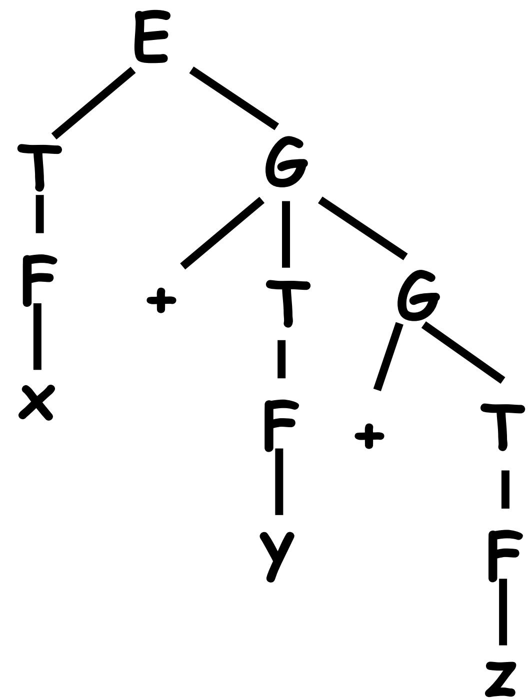
左再帰除去の問題点

- 除去後の文法に基づく構文解析木が元の文の論理的構造を反映しない

x+y+z



$E \to T \mid E+T$
$T \to F \mid T*F$
$F \to id \mid (E)$
に基づく解析木



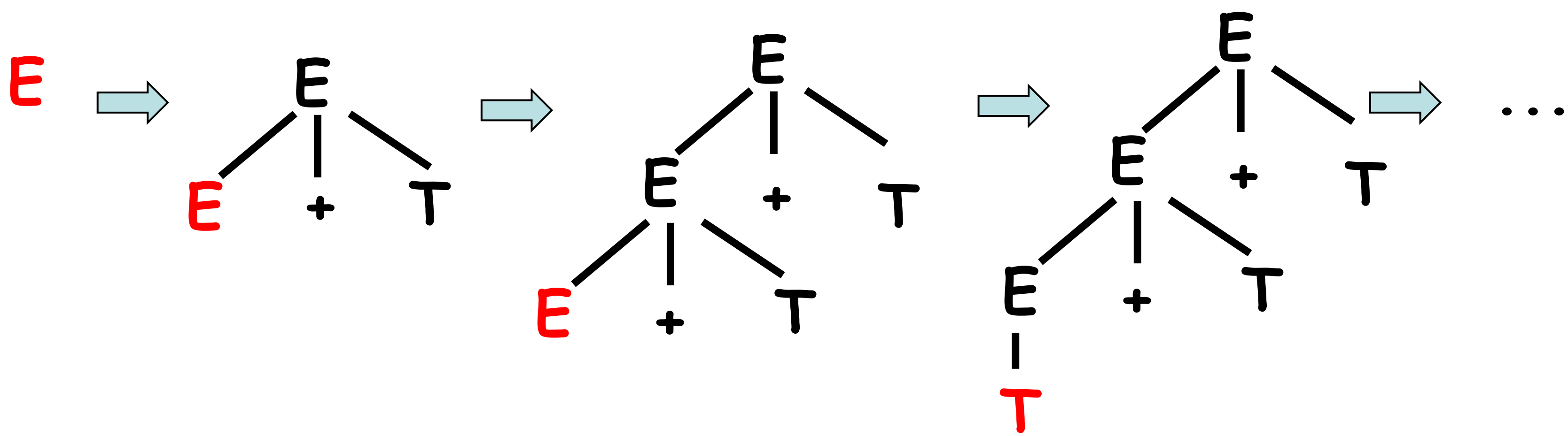
$E \to T \mid T\ G$
$G \to +T \mid +T\ G$
...
に基づく解析木

アウトライン

- 基本原則と例外
  （および文脈自由文法の復習）
- 文法の曖昧性
- 構文解析アルゴリズム
  - CKY法
  - トップダウン構文解析
    - ナイーブなアルゴリズム（バックトラックあり）
    - 先読みつき（LL(k)）

LL(k)

- Left-to-right parse, Leftmost-derivation with k-symbol lookahead
- トップダウン構文解析の一種
- 入力を左から右に走査(Left-to-right parse)
- 構文解析木を左から順に構成 (Leftmost-derivation)



LL(k)

- Left-to-right parse, Leftmost-derivation with k-symbol lookahead
- トップダウン構文解析の一種
- 入力を左から右に走査(Left-to-right parse)
- 構文解析木を左から順に構成 (Leftmost-derivation)
- k文字先読みしてどの生成規則を適用するかを判断（バックトラックは無し。適用する規則が一つに決まらない文法は扱わない）

LL(k)

- k文字先読みしてどの生成規則を適用するかを判断（バックトラックは無し。適用する規則が一つに決まらない文法は扱わない）

例：　S → AC | BD　　A → aa　　B → ab　...

　Sを認識中に、 S → AC とS → BDのどちらを用いるかを判断するには２文字先読みすればよい。

```
parseS() =
    match lookahead2() with
        "aa" -> parseA(); parseC()
      | "ab" -> parseB(); parseD()
      |  _ -> error()
```

LL(1)解析器の構成

(1) 各非終端記号Xについて以下を計算
（計算方法は後述）

- Nulls: $\{X \mid X \rightarrow^* \varepsilon\}$
  例：{G, H}
- FIRST(X): $\{a \mid X \rightarrow^* aw\}$
  (Xから生成される語の先頭文字の集合）
  例：S,E,T,F: {id, (}　G:{+, -}　H: {*}
- FOLLOW(X): $\{a \mid S \rightarrow^* wXaw'\}$
  (Xの次に続きうる終端記号の集合）
  例：S: {}　E,G: {), $}　T,H: {$,+,-,)}　F: {*,+,-,),$}

Running example:

$S \rightarrow E\$$　　$E \rightarrow T\ G$　　$G \rightarrow \varepsilon \mid +E \mid -E$

$T \rightarrow F\ H$　　$H \rightarrow \varepsilon \mid *T$　　$F \rightarrow id \mid (E)$

LL(1)解析器の構成

(1) 各非終端記号Xについて以下を計算

- Nulls: $\{X \mid X \rightarrow^* \varepsilon\}$
- FIRST(X): $\{a \mid X \rightarrow^* aw\}$
- FOLLOW(X): $\{a \mid S \rightarrow^* wXaw'\}$

用途：

現在認識しようとしている記号がX,
先読みトークンがcの場合、
規則 $X \rightarrow Y_1 \ldots Y_n$ を適用できるのは次のいずれか

(1) $Y_1, \ldots, Y_i \in$ Nulls かつ $c \in$ FIRST(Y_{i+1})

(2) $Y_1 \ldots Y_n \in$ Nullsかつ $c \in$ FOLLOW(X)

-> これらの条件が成り立つときのみ

$X \rightarrow Y_1 \ldots Y_n$ に従って解析

LL(1)解析器の構成

(1) 各非終端記号Xについて以下を計算

- Nulls: {X | X $\to^*$ ε }
- FIRST(X): {a | X $\to^*$ aw}
- FOLLOW(X): {a | S $\to^*$ wXaw'}

(2) 各非終端記号と先読み記号の組について
選択すべき遷移規則を表(LL(1)構文解析表）で表す

LL(1)解析器の構成

(2) LL(1)解析表の構成

	+	-	id	(	)	*	$
S							
E							
G							
T							
H							
E							

S → E$　　E → T G　　G → ε | +E | -E

T → F H　　H → ε | *T　　F → id | (E)

Nulls ={G, H}

FIRST S,E,T,F: {id, (}　G:{+, -}　H: {*}

FOLLOW S: {}　E,G: {), $}　T,H: {$,+,-,)}　F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)解析表の構成

	+	-	id	(	)	*	$
S							
E							
G							
T							
H							
E							

$S \rightarrow E\$$ $E \rightarrow T\ G$ $G \rightarrow \varepsilon \mid +E \mid -E$

$T \rightarrow F\ H$ $H \rightarrow \varepsilon \mid *T$ $F \rightarrow id \mid (E)$

Nulls ={G, H}

FIRST S,E,T,F: {id, (} G:{+, -} H: {*}

FOLLOW S: {} E,G: {), $} T,H: {$,+,-,)} F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)解析表の構成

	+	-	id	(	)	*	$
S							
E							
G							
T							
H							
E							

$S \to E\$$ $E \to T\ G$ $G \to \varepsilon \mid +E \mid -E$

$T \to F\ H$ $H \to \varepsilon \mid *T$ $F \to id \mid (E)$

Nulls ={G, H}

FIRST S,E,T,F: {id, (} G:{+, -} H: {*}

FOLLOW S: {} E,G: {), $} T,H: {$,+,-,)} F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)解析表の構成

	+	-	id	(	)	*	$
S							
E							
G							
T							
H							
E							

S → E$　　E → T G　　G → ε | +E | -E
T → F H　　H → ε | *T　　F → id | (E)
Nulls ={G, H}
FIRST S,E,T,F: {id, (}　　G:{+, -}　　H: {*}
FOLLOW S: {}　E,G: {), $}　T,H: {$,+,-,)}　F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)解析表の構成

	+	-	id	(	)	*	$
S							
E							
G							
T							
H							
E							

$S \rightarrow E\$$　　$E \rightarrow T\ G$　　$G \rightarrow \varepsilon \mid +E \mid -E$

$T \rightarrow F\ H$　　$H \rightarrow \varepsilon \mid *T$　　$F \rightarrow id \mid (E)$

Nulls ={G, H}

FIRST S,E,T,F: {id, (}　　G:{+, -}　　H: {*}

FOLLOW S: {}　E,G: {), $}　T,H: {$,+,-,)}　F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)解析表の構成

	+	-	id	(	)	*	$
S			E$				
E							
G							
T							
H							
E							

S → E$　　E → T G　　G → ε | +E | -E

T → F H　　H → ε | *T　　F → id | (E)

Nulls ={G, H}

FIRST S,E,T,F: {id, (}　　G:{+, -}　　H: {*}

FOLLOW S: {}　E,G: {), $}　T,H: {$,+,-,)}　F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)解析表の構成

	+	-	id	(	)	*	$
S			E$				
E							
G							
T							
H							
E							

S → E$　　E → T G　　G → ε | +E | -E

T → F H　　H → ε | *T　　F → id | (E)

Nulls ={G, H}

FIRST S,E,T,F: {id, (}　　G:{+, -}　　H: {*}

FOLLOW S: {}　E,G: {), $}　T,H: {$,+,-,)}　F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)解析表の構成

	+	-	id	(	)	*	$
S			E$	E$			
E							
G							
T							
H							
E							

S → E$ E → T G G → ε | +E | -E
T → F H H → ε | *T F → id | (E)
Nulls ={G, H}
FIRST S,E,T,F: {id, (} G:{+, -} H: {*}
FOLLOW S: {} E,G: {), $} T,H: {$,+,-,)} F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)解析表の構成

	+	-	id	(	)	*	$
S			E$	E$			
E							
G							
T							
H							
E							

S → E$ E → T G G → ε | +E | -E
T → F H H → ε | *T F → id | (E)
Nulls ={G, H}
FIRST S,E,T,F: {id, (} G:{+, -} H: {*}
FOLLOW S: {} E,G: {), $} T,H: {$,+,-,)} F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)解析表の構成

	+	-	id	(	)	*	$
S			E$	E$			
E							
G							
T							
H							
E							

S → E$　　E → T G　　G → ε | +E | -E
T → F H　　H → ε | *T　　F → id | (E)
Nulls ={G, H}
FIRST S,E,T,F: {id, (}　G:{+, -}　H: {*}
FOLLOW S: {}　E,G: {), $}　T,H: {$,+,-,)}　F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)解析表の構成

	+	-	id	(	)	*	$
S			E$	E$			
E							
G							
T							
H							
E							

S → E$　　E → T G　　G → ε | +E | -E
T → F H　　H → ε | *T　　F → id | (E)
Nulls ={G, H}
FIRST S,E,T,F: {id, (}　G:{+, -}　H: {*}
FOLLOW S: {}　E,G: {), $}　T,H: {$,+,-,)}　F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)解析表の構成

	+	-	id	(	)	*	$
S			E$	E$			
E			TG	TG			
G							
T							
H							
E							

S → E$　　E → T G　　G → ε | +E | -E
T → F H　　H → ε | *T　　F → id | (E)
Nulls ={G, H}
FIRST S,E,T,F: {id, (}　G:{+, -}　H: {*}
FOLLOW S: {}　E,G: {), $}　T,H: {$,+,-,)}　F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)解析表の構成

	+	-	id	(	)	*	$
S			E$	E$			
E			TG	TG			
G							
T							
H							
E							

S → E$ E → T G G → ε | +E | -E
T → F H H → ε | *T F → id | (E)
Nulls ={G, H}
FIRST S,E,T,F: {id, (} G:{+, -} H: {*}
FOLLOW S: {} E,G: {), $} T,H: {$,+,-,)} F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)解析表の構成

	+	-	id	(	)	*	$
S			E$	E$			
E			TG	TG			
G							
T							
H							
E							

S → E$　　E → T G　　G → ε | +E | -E
T → F H　　H → ε | *T　　F → id | (E)
Nulls ={G, H}
FIRST S,E,T,F: {id, (}　G:{+, -}　H: {*}
FOLLOW S: {}　E,G: {), $}　T,H: {$,+,-,)}　F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)解析表の構成

	+	-	id	(	)	*	$
S			E$	E$			
E			TG	TG			
G	+E	-E					
T							
H							
E							

S → E$　　E → T G　　G → ε | +E | -E

T → F H　　H → ε | *T　　F → id | (E)

Nulls ={G, H}

FIRST S,E,T,F: {id, (}　　G:{+, -}　　H: {*}

FOLLOW S: {}　E,G: {), $}　T,H: {$,+,-,)}　F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)解析表の構成

	+	-	id	(	)	*	$
S			E$	E$			
E			TG	TG			
G	+E	-E			ε		ε
T							
H							
E							

S → E$ E → T G G → ε | +E | -E
T → F H H → ε | *T F → id | (E)
Nulls ={G, H}
FIRST S,E,T,F: {id, (} G:{+, -} H: {*}
FOLLOW S: {} E,G: {), $} T,H: {$,+,-,)} F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)構文解析表の構成

	+	-	id	(	)	*	$
S			E$	E$			
E			TG	TG			
G	+E	-E			ε		ε
T			FH	FH			
H							
E							

$S \to E\$$　　$E \to T\ G$　　$G \to \varepsilon \mid +E \mid -E$

$T \to F\ H$　　$H \to \varepsilon \mid *T$　　$F \to id \mid (E)$

Nulls ={G, H}

FIRST S,E,T,F: {id, (}　G:{+, -}　H: {*}

FOLLOW S: {}　E,G: {), $}　T,H: {$,+,-,)}　F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)構文解析表の構成

	+	-	id	(	)	*	$
S			E$	E$			
E			TG	TG			
G	+E	-E			ε		ε
T			FH	FH			
H	ε	ε			ε	*T	ε
E							

S → E$　　E → T G　　G → ε | +E | -E
T → F H　　H → ε | *T　　F → id | (E)
Nulls ={G, H}
FIRST S,E,T,F: {id, (}　　G:{+, -}　　H: {*}
FOLLOW S: {}　E,G: {), $}　T,H: {$,+,-,)}　F: {*,+,-,),$}

LL(1)解析器の構成

(2) LL(1)構文解析表の構成

	+	-	id	(	)	*	$
S			E$	E$			
E			TG	TG			
G	+E	-E			ε		ε
T			FH	FH			
H	ε	ε			ε	*T	ε
E			id	(E)			

S → E$ E → T G G → ε | +E | -E
T → F H H → ε | *T F → id | (E)
Nulls ={G, H}
FIRST S,E,T,F: {id, (} G:{+, -} H: {*}
FOLLOW S: {} E,G: {), $} T,H: {$,+,-,)} F: {*,+,-,),$}

LL(1)解析器の構成

(1) 各非終端記号Xについて以下を計算
- Nulls: $\{X \mid X \rightarrow^* \varepsilon\}$
- FIRST(X): $\{a \mid X \rightarrow^* aw\}$
- FOLLOW(X): $\{a \mid S \rightarrow^* wXaw'\}$

(2) LL(1)構文解析表の構成

(3) 構文解析表を見ながら解析するコードを生成

LL(1)解析器の構成

(3) 構文解析表を見ながら解析するコードを生成

```
parse(X) =
    if X is a terminal then
        if next()=X then PTterm(X) else error()
    else (* non-terminal *)
        match lookup_LLtable(X, lookahead()) with
            Some(Y1...Yn) ->
                let p1 = parse(Y1) in
                 ...
                let pn = parse(Yn) in
                    PTnt(X, [p1;...;pn])
         | None -> error()
```

LL(1)で扱えない例

S → E$　　E → T | T+E
T → F H　　H → ε | *T　　F → id | (E)
Nulls ={H}
FIRST S,E,T,F: {id, (}　H: {*}
FOLLOW S: {}　E: {), $}　T,H: {$,+,)}　F: {*,+,),$}

	+	id	(	)	*	$
S		E$	E$			
E		T, T+E	T, T+E			
T		FH	FH			
H	ε			ε	*T	ε

衝突
(conflict)

LL(k)

- k文字先読みしてLL(1)と同様の処理
- LL(k)構文解析表の列には長さkの文字列を記入
  - 利点：より広い範囲の文法を扱える
  - 欠点：構文解析表が大きくなりすぎる
    （列のサイズが$O(n^k)$）

LL(k)文法とLL(k)言語

- LL(k)文法
  - LL(k)で扱える文法
    （LL(k)構文解析表に衝突がない）
- LL(k)言語
  - あるLL(k)文法で生成できる言語

$S \to E\$$　　$E \to T \mid T{+}E$

$T \to F\ H$　　$H \to \varepsilon \mid {*}T$　　$F \to id \mid (E)$

はLL(1)文法でないが、生成される言語はLL(1)言語
(Eの規則を$E \to TG$, $G \to \varepsilon \mid {+}E$ とすれば
LL(1)文法なので)

Null, First, Follow の計算

- 原則
  - 規則から集合Nulls, FIRST(X), FOLLOW(X)に関する制約を生成
  - 制約を解く（最小解を求める）

Nulls: $\{X \mid X \to^* \varepsilon\}$
FIRST(X): $\{a \mid X \to^* aw\}$
（Xから生成される語の先頭文字の集合）
FOLLOW(X): $\{a \mid S \to^* wXaw'\}$
（Xの次に続きうる終端記号の集合）

Nulls ($\{X \mid X \to^* \varepsilon\}$)の求め方

制約生成：各書き換え規則 $X \to Y_1 \ldots Y_n$
(Y_iは終端または非終端記号）について以下の制約を生成

$$Y_1 \in \text{Nulls} \wedge \ldots \wedge Y_n \in \text{Nulls} \Rightarrow X \in \text{Nulls}$$

Running example:

$C \to d$	$B \to$	$A \to B$
$C \to A\ B\ C$	$B \to c$	$A \to a$

Nulls ($\{X \mid X \to^* \varepsilon\}$)の求め方

制約生成：各書き換え規則 $X \to Y_1 \dots Y_n$
（Y_iは終端または非終端記号）について以下の制約を生成

$$Y_1 \in \mathrm{Nulls} \wedge \dots \wedge Y_n \in \mathrm{Nulls} \Rightarrow X \in \mathrm{Nulls}$$

Running example:

$C \to d$	$B \to$	$A \to B$
$C \to A\ B\ C$	$B \to c$	$A \to a$

$d \in \mathrm{Nulls} \Rightarrow C \in \mathrm{Nulls}$

Nulls ($\{X \mid X \rightarrow^* \varepsilon\}$)の求め方

制約生成：各書き換え規則 $X \rightarrow Y_1 \ldots Y_n$

（Y_i**は終端または非終端記号）について以下の制約を生成**

$$Y_1 \in \text{Nulls} \wedge \ldots \wedge Y_n \in \text{Nulls} \Rightarrow X \in \text{Nulls}$$

Running example:

$C \rightarrow d$	$B \rightarrow$	$A \rightarrow B$
$C \rightarrow A\ B\ C$	$B \rightarrow c$	$A \rightarrow a$

$d \in \text{Nulls} \Rightarrow C \in \text{Nulls}$　　$\text{true} \Rightarrow B \in \text{Nulls}$

Nulls ($\{X \mid X \to^* \varepsilon\}$)の求め方

制約生成：各書き換え規則 $X \to Y_1 \ldots Y_n$
（Y_i**は終端または非終端記号）について以下の制約を生成**

$$Y_1 \in \text{Nulls} \wedge \ldots \wedge Y_n \in \text{Nulls} \Rightarrow X \in \text{Nulls}$$

Running example:

$C \to d$	$B \to$	$A \to B$
$C \to A\ B\ C$	$B \to c$	$A \to a$

$d \in \text{Nulls} \Rightarrow C \in \text{Nulls}$　　$\text{true} \Rightarrow B \in \text{Nulls}$　　$B \in \text{Nulls} \Rightarrow A \in \text{Nulls}$

Nulls ($\{X \mid X \to^* \varepsilon\}$)の求め方

制約生成：各書き換え規則 $X \to Y_1 \ldots Y_n$
（Y_iは終端または非終端記号）について以下の制約を生成

$$Y_1 \in \text{Nulls} \wedge \ldots \wedge Y_n \in \text{Nulls} \Rightarrow X \in \text{Nulls}$$

Running example:

$C \to d$ $\quad B \to$ $\quad A \to B$

$C \to A\ B\ C$ $\quad B \to c$ $\quad A \to a$

$d \in \text{Nulls} \Rightarrow C \in \text{Nulls}$ $\quad \text{true} \Rightarrow B \in \text{Nulls}$ $\quad B \in \text{Nulls} \Rightarrow A \in \text{Nulls}$

$A \in \text{Nulls} \wedge B \in \text{Nulls} \wedge C \in \text{Nulls} \Rightarrow C \in \text{Nulls}$

Nulls ($\{X \mid X \rightarrow^* \varepsilon\}$)の求め方

制約生成：各書き換え規則 $X \rightarrow Y_1 \ldots Y_n$
（Y_i**は終端または非終端記号）について以下の制約を生成**

$$Y_1 \in Nulls \wedge \ldots \wedge Y_n \in Nulls \Rightarrow X \in Nulls$$

Running example:

$C \rightarrow d$　　$B \rightarrow$　　$A \rightarrow B$
$C \rightarrow A\ B\ C$　　$B \rightarrow c$　　$A \rightarrow a$

⇓

$d \in Nulls \Rightarrow C \in Nulls$　　$true \Rightarrow B \in Nulls$　　$B \in Nulls \Rightarrow A \in Nulls$
$A \in Nulls \wedge B \in Nulls \wedge C \in Nulls \Rightarrow C \in Nulls$　　$c \in Nulls \Rightarrow B \in Nulls$
$a \in Nulls \Rightarrow A \in Nulls$

制約解消：Nulls = {} **と初期化し、**
「…$\Rightarrow X \in Nulls$ **の左辺が真であればXをNullsに追加」**
を繰り返す

Nulls = {　　　　　　　　　　}

Nulls ($\{X \mid X \rightarrow^* \varepsilon\}$)の求め方

制約生成：各書き換え規則 $X \rightarrow Y_1 \ldots Y_n$
(Y_i**は終端または非終端記号）について以下の制約を生成**

$$Y_1 \in \mathrm{Nulls} \wedge \ldots \wedge Y_n \in \mathrm{Nulls} \Rightarrow X \in \mathrm{Nulls}$$

Running example:

$C \rightarrow d$ $\quad$ $B \rightarrow$ $\quad$ $A \rightarrow B$

$C \rightarrow A\ B\ C$ $\quad$ $B \rightarrow c$ $\quad$ $A \rightarrow a$

$d \in \mathrm{Nulls} \Rightarrow C \in \mathrm{Nulls}$ $\quad$ $\mathrm{true} \Rightarrow B \in \mathrm{Nulls}$ $\quad$ $B \in \mathrm{Nulls} \Rightarrow A \in \mathrm{Nulls}$

$A \in \mathrm{Nulls} \wedge B \in \mathrm{Nulls} \wedge C \in \mathrm{Nulls} \Rightarrow C \in \mathrm{Nulls}$ $\quad$ $c \in \mathrm{Nulls} \Rightarrow B \in \mathrm{Nulls}$

$a \in \mathrm{Nulls} \Rightarrow A \in \mathrm{Nulls}$

制約解消：Nulls = {} **と初期化し、**
「$\ldots \Rightarrow X \in \mathrm{Nulls}$ **の左辺が真であればXをNullsに追加」**
を繰り返す

Nulls = { B, }

Nulls ({X | X $\to^*$ ε })の求め方

制約生成：各書き換え規則 X $\to$ $Y_1 \ldots Y_n$
(Y_i**は終端または非終端記号）について以下の制約を生成**

$$Y_1 \in \mathrm{Nulls} \wedge \ldots \wedge Y_n \in \mathrm{Nulls} \Rightarrow X \in \mathrm{Nulls}$$

Running example:

C → d　　B →　　A → B
C → A B C　　B → c　　A → a

d∈Nulls ⇒ C∈ Nulls　　true ⇒ B∈Nulls　　B∈Nulls ⇒A∈Nulls
A∈Nulls∧B∈Nulls∧C∈Nulls ⇒C∈Nulls　　c∈Nulls⇒B∈ Nulls
a∈Nulls⇒A∈ Nulls

制約解消：Nulls = {} **と初期化し、**
「…⇒ X ∈ Nulls **の左辺が真であれば**X**を**Nulls**に追加**」
を繰り返す

Nulls = { B, A }

FIRST(X)= $\{a \mid X \rightarrow^* aw\}$の求め方

制約生成：各書き換え規則 $X \rightarrow Y_1 \ldots Y_n$およびk=1,…,n
(Y_iは終端または非終端記号）について以下の制約を生成

$$Y_1 \in \text{Nulls} \wedge \ldots \wedge Y_{k-1} \in \text{Nulls} \Rightarrow \text{FIRST}(X) \supseteq \text{FIRST}(Y_k)$$

(ただしY_kが終端端記号aの場合はFIRST(Y_k)={a}）

$C \rightarrow d$	$B \rightarrow$	$A \rightarrow B$
$C \rightarrow A\ B\ C$	$B \rightarrow c$	$A \rightarrow a$

Nulls = {A,B}

FIRST(X)= {a | X →* aw}の求め方

制約生成：各書き換え規則 $X \rightarrow Y_1 \ldots Y_n$ および $k=1,\ldots,n$（Y_iは終端または非終端記号）について以下の制約を生成

$$Y_1 \in Nulls \wedge \ldots \wedge Y_{k-1} \in Nulls \Rightarrow FIRST(X) \supseteq FIRST(Y_k)$$

(ただしY_kが終端端記号aの場合はFIRST(Y_k)={a})

C → d	B →	A → B
C → A B C	B → c	A → a

Nulls = {A,B}

FIRST(C) ⊇ {d}

FIRST(X)= {a | X →* aw}の求め方

制約生成：各書き換え規則 $X \rightarrow Y_1 \ldots Y_n$ および$k=1,\ldots,n$
(Y_iは終端または非終端記号）について以下の制約を生成

$$Y_1 \in Nulls \wedge \ldots \wedge Y_{k-1} \in Nulls \Rightarrow FIRST(X) \supseteq FIRST(Y_k)$$

(ただしY_kが終端端記号aの場合は$FIRST(Y_k)=\{a\}$)

$C \rightarrow d$	$B \rightarrow$	$A \rightarrow B$
$C \rightarrow A\ B\ C$	$B \rightarrow c$	$A \rightarrow a$

$Nulls = \{A, B\}$

$FIRST(C) \supseteq \{d\}$ $\quad$ $FIRST(A) \supseteq FIRST(B)$

FIRST(X)= {a | X →* aw}の求め方

制約生成：各書き換え規則 $X \to Y_1 \ldots Y_n$ **および**$k=1,\ldots,n$
(Y_i**は終端または非終端記号）について以下の制約を生成**

$$Y_1 \in \mathrm{Nulls} \wedge \ldots \wedge Y_{k-1} \in \mathrm{Nulls} \Rightarrow \mathrm{FIRST}(X) \supseteq \mathrm{FIRST}(Y_k)$$

(ただしY_k**が終端端記号**a**の場合は**$\mathrm{FIRST}(Y_k)=\{a\}$**）**

C → d	B →	A → B
C → A B C	B → c	A → a

Nulls = {A,B}

FIRST(C) ⊇ {d}　FIRST(A) ⊇ FIRST(B)　FIRST(C) ⊇ FIRST(A)
FIRST(C) ⊇ FIRST(B)　FIRST(C) ⊇ FIRST(C)

FIRST(X)= {a | X →* aw}の求め方

制約生成：各書き換え規則 $X \to Y_1 \ldots Y_n$**および**$k=1,\ldots,n$
(Y_i**は終端または非終端記号）について以下の制約を生成**

$$Y_1 \in \text{Nulls} \wedge \ldots \wedge Y_{k-1} \in \text{Nulls} \Rightarrow \text{FIRST}(X) \supseteq \text{FIRST}(Y_k)$$

(**ただし**Y_k**が終端端記号**a**の場合は**$\text{FIRST}(Y_k)=\{a\}$)

$C \to d$	$B \to$	$A \to B$
$C \to A\ B\ C$	$B \to c$	$A \to a$

Nulls = {A,B}

$\text{FIRST}(C) \supseteq \{d\}$　$\text{FIRST}(A) \supseteq \text{FIRST}(B)$　$\text{FIRST}(C) \supseteq \text{FIRST}(A)$
$\text{FIRST}(C) \supseteq \text{FIRST}(B)$　$\text{FIRST}(C) \supseteq \text{FIRST}(C)$
$\text{FIRST}(B) \supseteq \{c\}$　$\text{FIRST}(A) \supseteq \{a\}$

制約解消：FIRST(X) = {} **と初期化し、**
すべての制約が満たされるまで要素を追加

FIRST(A) = {　　　　　}
FIRST(B) = {　　　　　}　FIRST(C) = {　　　　}

FIRST(X)= {a | X →* aw}の求め方

制約生成：各書き換え規則 $X \to Y_1 \ldots Y_n$ および $k=1,\ldots,n$
(Y_iは終端または非終端記号）について以下の制約を生成

$$Y_1 \in \text{Nulls} \wedge \ldots \wedge Y_{k-1} \in \text{Nulls} \Rightarrow \text{FIRST}(X) \supseteq \text{FIRST}(Y_k)$$

(ただしY_k終端記号aの場合は$\text{FIRST}(Y_k)=\{a\}$)

C → d	B →	A → B
C → A B C	B → c	A → a

⇩

$\text{FIRST}(C) \supseteq \{d\}$　$\text{FIRST}(A) \supseteq \text{FIRST}(B)$　$\text{FIRST}(C) \supseteq \text{FIRST}(A)$
$\text{FIRST}(C) \supseteq \text{FIRST}(B)$　$\text{FIRST}(C) \supseteq \text{FIRST}(C)$
$\text{FIRST}(B) \supseteq \{c\}$　$\text{FIRST}(A) \supseteq \{a\}$

制約解消：FIRST(X) = {} と初期化し、
すべての制約が満たされるまで要素を追加

FIRST(A) = { a }
FIRST(B) = { c }　FIRST(C) = { d }

FIRST(X)= {a | X →* aw}の求め方

制約生成：各書き換え規則 $X \to Y_1 \ldots Y_n$ および$k=1,\ldots,n$
(Y_iは終端または非終端記号）について以下の制約を生成

$$Y_1 \in \text{Nulls} \wedge \ldots \wedge Y_{k-1} \in \text{Nulls} \Rightarrow \text{FIRST}(X) \supseteq \text{FIRST}(Y_k)$$

(ただしY_k終端記号aの場合は$\text{FIRST}(Y_k)=\{a\}$)

C → d	B →	A → B
C → A B C	B → c	A → a

$\text{FIRST}(C) \supseteq \{d\}$　　$\text{FIRST}(A) \supseteq \text{FIRST}(B)$　　$\text{FIRST}(C) \supseteq \text{FIRST}(A)$
$\text{FIRST}(C) \supseteq \text{FIRST}(B)$　　$\text{FIRST}(C) \supseteq \text{FIRST}(C)$
$\text{FIRST}(B) \supseteq \{c\}$　　$\text{FIRST}(A) \supseteq \{a\}$

制約解消：FIRST(X) = {} と初期化し、
すべての制約が満たされるまで要素を追加

FIRST(A) = { a c }
FIRST(B) = { c }　FIRST(C) = { d }

$\mathrm{FIRST}(X) = \{a \mid X \to^* aw\}$の求め方

制約生成：各書き換え規則 $X \to Y_1 \dots Y_n$**および**$k=1,\dots,n$
(Y_i**は終端または非終端記号）について以下の制約を生成**

$$Y_1 \in \mathrm{Nulls} \wedge \dots \wedge Y_{k-1} \in \mathrm{Nulls} \Rightarrow \mathrm{FIRST}(X) \supseteq \mathrm{FIRST}(Y_k)$$

(**ただし**Y_k**終端記号**a**の場合は**$\mathrm{FIRST}(Y_k)=\{a\}$)

$C \to d$	$B \to$	$A \to B$
$C \to A\ B\ C$	$B \to c$	$A \to a$

⇩

$\mathrm{FIRST}(C) \supseteq \{d\}$　$\mathrm{FIRST}(A) \supseteq \mathrm{FIRST}(B)$　$\mathrm{FIRST}(C) \supseteq \mathrm{FIRST}(A)$
$\mathrm{FIRST}(C) \supseteq \mathrm{FIRST}(B)$　$\mathrm{FIRST}(C) \supseteq \mathrm{FIRST}(C)$
$\mathrm{FIRST}(B) \supseteq \{c\}$　$\mathrm{FIRST}(A) \supseteq \{a\}$

制約解消：$\mathrm{FIRST}(X) = \{\}$ **と初期化し、**
すべての制約が満たされるまで要素を追加

$\mathrm{FIRST}(A) = \{\ a\quad c\ \}$
$\mathrm{FIRST}(B) = \{\ c\ \}$　$\mathrm{FIRST}(C) = \{\ d\quad a\quad c\ \}$

FOLLOW(X)= {a | S →* αXaβ}の求め方

制約生成：各書き換え規則 X → $Y_1 \ldots Y_n$
(Y_iは終端または非終端記号）について以下の制約を生成

$Y_{k+1} \in \mathrm{Nulls} \wedge \ldots \wedge Y_{m-1} \in \mathrm{Nulls} \Rightarrow \mathrm{FOLLOW}(Y_k) \supseteq \mathrm{FIRST}(Y_m)$ $(1 \leq k < m \leq n)$

$Y_{k+1} \in \mathrm{Nulls} \wedge \ldots \wedge Y_n \in \mathrm{Nulls} \Rightarrow \mathrm{FOLLOW}(Y_k) \supseteq \mathrm{FOLLOW}(X)$ $(1 \leq k \leq n)$

C → d	**B →**	**A → B**
C → A B C	**B → c**	**A → a**

Nulls = {A,B}

FOLLOW(X)= {a | S →* αXaβ}の求め方

制約生成：各書き換え規則 $X \to Y_1 \ldots Y_n$
（Y_i**は終端または非終端記号）について以下の制約を生成**

$Y_{k+1} \in \mathrm{Nulls} \wedge \ldots \wedge Y_{m-1} \in \mathrm{Nulls} \Rightarrow \mathrm{FOLLOW}(Y_k) \supseteq \mathrm{FIRST}(Y_m)$ $(1 \le k < m \le n)$

$Y_{k+1} \in \mathrm{Nulls} \wedge \ldots \wedge Y_n \in \mathrm{Nulls} \Rightarrow \mathrm{FOLLOW}(Y_k) \supseteq \mathrm{FOLLOW}(X)$ $(1 \le k \le n)$

$C \to d$	$B \to$	$A \to B$
$C \to A\ B\ C$	$B \to c$	$A \to a$

Nulls = {A,B}

FOLLOW(B) ⊇ FOLLOW(A)

FOLLOW(X)= $\{a \mid S \rightarrow^* \alpha X a \beta\}$の求め方

制約生成：各書き換え規則 $X \rightarrow Y_1 \ldots Y_n$
(Y_i**は終端または非終端記号）について以下の制約を生成**

$Y_{k+1} \in Nulls \wedge \ldots \wedge Y_{m-1} \in Nulls \Rightarrow FOLLOW(Y_k) \supseteq FIRST(Y_m) \ (1 \le k < m \le n)$

$Y_{k+1} \in Nulls \wedge \ldots \wedge Y_n \in Nulls \Rightarrow FOLLOW(Y_k) \supseteq FOLLOW(X) \ (1 \le k \le n)$

$C \rightarrow d$	$B \rightarrow$	$A \rightarrow B$
$C \rightarrow A\ B\ C$	$B \rightarrow c$	$A \rightarrow a$

Nulls = {A,B}

FOLLOW(B) $\supseteq$ FOLLOW(A)   FOLLOW(A)$\supseteq$ FIRST(B)={c}

FOLLOW(X)= $\{a \mid S \rightarrow^* \alpha X a \beta\}$の求め方

制約生成：各書き換え規則 $X \rightarrow Y_1 \ldots Y_n$
(Y_iは終端または非終端記号）について以下の制約を生成

$Y_{k+1} \in Nulls \wedge \ldots \wedge Y_{m-1} \in Nulls \Rightarrow FOLLOW(Y_k) \supseteq FIRST(Y_m)$ $(1 \leq k < m \leq n)$

$Y_{k+1} \in Nulls \wedge \ldots \wedge Y_n \in Nulls \Rightarrow FOLLOW(Y_k) \supseteq FOLLOW(X)$ $(1 \leq k \leq n)$

$C \rightarrow d$ $\quad B \rightarrow$ $\quad A \rightarrow B$
$C \rightarrow A\ B\ C$ $\quad B \rightarrow c$ $\quad A \rightarrow a$

Nulls = {A,B}

FOLLOW(B) ⊇ FOLLOW(A) $\quad$ FOLLOW(A) ⊇ FIRST(B)={c}
FOLLOW(A) ⊇ FIRST(C)={a,c,d}

FOLLOW(X)= $\{a \mid S \to^* \alpha X a \beta\}$の求め方

制約生成：各書き換え規則 $X \to Y_1 \ldots Y_n$
(Y_i**は終端または非終端記号）について以下の制約を生成**

$Y_{k+1} \in Nulls \wedge \ldots \wedge Y_{m-1} \in Nulls \Rightarrow FOLLOW(Y_k) \supseteq FIRST(Y_m)$ $(1 \leq k < m \leq n)$

$Y_{k+1} \in Nulls \wedge \ldots \wedge Y_n \in Nulls \Rightarrow FOLLOW(Y_k) \supseteq FOLLOW(X)$ $(1 \leq k \leq n)$

$C \to d$	$B \to$	$A \to B$
$C \to A\ B\ C$	$B \to c$	$A \to a$

⇓ Nulls = {A,B}

FOLLOW(B) ⊇ FOLLOW(A)　FOLLOW(A) ⊇ FIRST(B)={c}
FOLLOW(A) ⊇ FIRST(C)={a,c,d}　FOLLOW(B) ⊇ FIRST(C)={a,c,d}

FOLLOW(X)= {a | S $\rightarrow^*$ αXaβ}の求め方

制約生成：各書き換え規則 $X \rightarrow Y_1 \ldots Y_n$
（Y_iは終端または非終端記号）について以下の制約を生成

$Y_{k+1} \in \mathrm{Nulls} \wedge \ldots \wedge Y_{m-1} \in \mathrm{Nulls} \Rightarrow \mathrm{FOLLOW}(Y_k) \supseteq \mathrm{FIRST}(Y_m) \quad (1 \leq k < m \leq n)$

$Y_{k+1} \in \mathrm{Nulls} \wedge \ldots \wedge Y_n \in \mathrm{Nulls} \Rightarrow \mathrm{FOLLOW}(Y_k) \supseteq \mathrm{FOLLOW}(X) \quad (1 \leq k \leq n)$

C → d	B →	A → B
C → A B C	B → c	A → a

Nulls = {A,B}

FOLLOW(B) ⊇ FOLLOW(A) FOLLOW(A) ⊇ FIRST(B)={c}
FOLLOW(A) ⊇ FIRST(C)={a,c,d} FOLLOW(B) ⊇ FIRST(C)={a,c,d}
FOLLOW(C) ⊇ FOLLOW(C)

制約解消：FOLLOW(X) = {} **と初期化し、**
すべての制約が満たされるまで要素を追加

FOLLOW(A) = {   }
FOLLOW(B) = {   } FOLLOW(C) = {   }

FOLLOW(X)= $\{a \mid S \rightarrow^* \alpha X a \beta\}$の求め方

制約生成：各書き換え規則 $X \rightarrow Y_1 \dots Y_n$
(Y_iは終端または非終端記号）について以下の制約を生成

$Y_{k+1} \in \mathrm{Nulls} \wedge \dots \wedge Y_{m-1} \in \mathrm{Nulls} \Rightarrow \mathrm{FOLLOW}(Y_k) \supseteq \mathrm{FIRST}(Y_m) \quad (1 \leq k < m \leq n)$

$Y_{k+1} \in \mathrm{Nulls} \wedge \dots \wedge Y_n \in \mathrm{Nulls} \Rightarrow \mathrm{FOLLOW}(Y_k) \supseteq \mathrm{FOLLOW}(X) \quad (1 \leq k \leq n)$

$C \rightarrow d$ $B \rightarrow$ $A \rightarrow B$
$C \rightarrow A\ B\ C$ $B \rightarrow c$ $A \rightarrow a$

FOLLOW(B) ⊇ FOLLOW(A)   FOLLOW(A)⊇ FIRST(B)={c}
FOLLOW(A)⊇ FIRST(C)={a,c,d}   FOLLOW(B) ⊇ FIRST(C)={a,c,d}
FOLLOW(C) ⊇ FOLLOW(C)

制約解消：FOLLOW(X) = {} **と初期化し、**
すべての制約が満たされるまで要素を追加

FOLLOW(A) = { a c d }
FOLLOW(B) = { a c d }   FOLLOW(C) = { }

一般的な制約解消法

不動点定理

- 仮定：
  - $(S, \leq)$: 半順序集合
  - Sの最小元 $\perp \in S$が存在
  - 狭義無限増加列 $\perp < x_1 < x_2 < \ldots$は存在しない
  - $f \in S \to S$ は単調関数
    (すなわち$\forall x,y.\ x \leq y \Rightarrow f(x) \leq f(y)$)
- 定理：ある$n \in Nat$が存在し、以下が成り立つ

  (i) $f^n(\perp) = f^{n+1}(\perp)$

  (ii) $x = f^n(\perp)$ は、 $f(x) \leq x$の最小解

不動点定理（証明）

- 定理：ある$n \in Nat$が存在し、以下が成り立つ

  (i) $f^n(\perp) = f^{n+1}(\perp)$

  (ii) $x = f^n(\perp)$ は、 $f(x) \leq x$の最小解

- 証明

(i) $\perp$の最小性より$\perp \leq f(\perp)$

fの単調性より任意のkについて$f^k(\perp) \leq f^{k+1}(\perp)$

よって、 $\perp \leq f(\perp) \leq f^2(\perp) \leq f^3(\perp) \leq \ldots$ は（広義）単調増加列。

狭義無限増加列は存在しないので、あるnについて$f^n(\perp) = f^{n+1}(\perp)$。

(ii) $f(x) \leq x$であると仮定する。$\perp$の最小性より$\perp \leq x$。

fの単調性より$f^n(\perp) \leq f^n(x)$。

$f(x) \leq x$とfの単調性より、 $f^n(x) \leq f^{n-1}(x) \leq \ldots \leq f(x) \leq x$。

よって、$f^n(\perp) \leq x$

仮定

$(S, \leq)$:半順序集合　　Sの最小元 $\perp \in S$が存在

狭義無限増加列 $\perp < x_1 < x_2 < \ldots$は存在しない

$f \in S \rightarrow S$ は単調関数

制約解消アルゴリズム

$x \geq f(x)$ の最小解を求めるアルゴリズム：

```
x := ⊥;
while (f(x)≠ x) do x := f(x);
return x
```

仮定
$(S, \leq)$:半順序集合　　Sの最小元 $\perp \in S$が存在
狭義無限増加列 $\perp < x_1 < x_2 < \ldots$は存在しない
$f \in S \to S$ は単調関数

例：Nullsの場合

$d \in Nulls \Rightarrow C \in Nulls$ 　 $true \Rightarrow B \in Nulls$ 　 $B \in Nulls \Rightarrow A \in Nulls$
$A \in Nulls \wedge B \in Nulls \wedge C \in Nulls \Rightarrow C \in Nulls$ 　 $c \in Nulls \Rightarrow B \in Nulls$
$a \in Nulls \Rightarrow A \in Nulls$

$\Downarrow$

$Nulls \supseteq f(Nulls)$ where:
$f(x) = (d \in x \Rightarrow \{C\}) \cup \{B\} \cup (B \in x \Rightarrow \{A\})$
$\cup\ (A \in x \wedge B \in x \wedge C \in x \Rightarrow \{C\}) \cup (c \in x \Rightarrow \{B\}) \cup (a \in x \Rightarrow \{A\})$
（ただし $b \Rightarrow S$ はbが真ならS，偽なら$\emptyset$ とする）
fは、$2^{\{A,B,C\}}$上の単調関数

$f(\emptyset) = \{B\}$ 　 $f^2(\emptyset) = f(\{B\}) = \{A,B\}$

$f^3(\emptyset) = f(\{A,B\}) = \{A,B\} = f^2(\emptyset)$

よって$Nulls = \{A,B\}$

$x \geq f(x)$ の最小解を求めるアルゴリズム：

```
x := ⊥;
while (f(x)≠ x) do x := f(x);
return x
```

例：First(X)の場合

$FIRST(C) \supseteq \{d\}$　$FIRST(A) \supseteq FIRST(B)$　$FIRST(C) \supseteq FIRST(A)$
$FIRST(C) \supseteq FIRST(B)$　$FIRST(C) \supseteq FIRST(C)$
$FIRST(B) \supseteq \{c\}$　$FIRST(A) \supseteq \{a\}$

$(FIRST(A), FIRST(B), FIRST(C)) \geq f(FIRST(A), FIRST(B), FIRST(C))$
where:
$f(x_A, x_B, x_C) =$

$x \geq f(x)$　の最小解を求めるアルゴリズム：

```
x := ⊥;
while (f(x)≠ x) do x := f(x);
return x
```

例：First(X)の場合

FIRST(C) ⊇ {d}　FIRST(A) ⊇ FIRST(B)　FIRST(C) ⊇ FIRST(A)
FIRST(C) ⊇ FIRST(B)　FIRST(C) ⊇ FIRST(C)
FIRST(B) ⊇ {c}　FIRST(A) ⊇ {a}

⇩

(FIRST(A),FIRST(B),FIRST(C)) ≥ f(FIRST(A),FIRST(B),FIRST(C))
where:

$f(x_A, x_B, x_C) = (x_B \cup \{a\},$

$x \geq f(x)$　の最小解を求めるアルゴリズム：

```
x := ⊥;
while (f(x)≠ x) do x := f(x);
return x
```

例：First(X)の場合

$FIRST(C) \supseteq \{d\}$ $FIRST(A) \supseteq FIRST(B)$ $FIRST(C) \supseteq FIRST(A)$
$FIRST(C) \supseteq FIRST(B)$ $FIRST(C) \supseteq FIRST(C)$
$FIRST(B) \supseteq \{c\}$ $FIRST(A) \supseteq \{a\}$

⇩

$(FIRST(A), FIRST(B), FIRST(C)) \geq f(FIRST(A), FIRST(B), FIRST(C))$
where:
$f(x_A, x_B, x_C) = (x_B \cup \{a\}, \{c\},$

$x \geq f(x)$ の最小解を求めるアルゴリズム：

```
x := ⊥;
while (f(x)≠ x) do x := f(x);
return x
```

例：First(X)の場合

$\mathrm{FIRST}(C) \supseteq \{d\}$　$\mathrm{FIRST}(A) \supseteq \mathrm{FIRST}(B)$　$\mathrm{FIRST}(C) \supseteq \mathrm{FIRST}(A)$
$\mathrm{FIRST}(C) \supseteq \mathrm{FIRST}(B)$　$\mathrm{FIRST}(C) \supseteq \mathrm{FIRST}(C)$
$\mathrm{FIRST}(B) \supseteq \{c\}$　$\mathrm{FIRST}(A) \supseteq \{a\}$

⇩

$(\mathrm{FIRST}(A),\mathrm{FIRST}(B),\mathrm{FIRST}(C)) \geq f(\mathrm{FIRST}(A),\mathrm{FIRST}(B),\mathrm{FIRST}(C))$

where:

$f(x_A,x_B,x_C) = (x_B \cup \{a\}, \{c\}, \{d\} \cup x_A \cup x_B \cup x_C)$

fは、$2^{\{a,c,d\}} \times 2^{\{a,c,d\}} \times 2^{\{a,c,d\}}$上の単調関数

例：First(X)の場合

$FIRST(C) \supseteq \{d\}$　　$FIRST(A) \supseteq FIRST(B)$　　$FIRST(C) \supseteq FIRST(A)$
$FIRST(C) \supseteq FIRST(B)$　　$FIRST(C) \supseteq FIRST(C)$
$FIRST(B) \supseteq \{c\}$　　$FIRST(A) \supseteq \{a\}$

⇩

$(FIRST(A),FIRST(B),FIRST(C)) \geq f(FIRST(A),FIRST(B),FIRST(C))$
where:

$$f(x_A,x_B,x_C) = (x_B \cup \{a\},\ \{c\},\ \{d\} \cup x_A \cup x_B \cup x_C)$$

fは、$2^{\{a,c,d\}} \times 2^{\{a,c,d\}} \times 2^{\{a,c,d\}}$上の単調関数

$$f(\emptyset,\ \emptyset,\ \emptyset) = (\{a\},\{c\},\{d\})$$

$$f^2(\emptyset,\ \emptyset,\ \emptyset) = f(\{a\},\{c\},\{d\}) = (\{a,c\},\ \{c\},\ \{a,c,d\})$$

$$\begin{aligned} f^3(\emptyset,\ \emptyset,\ \emptyset) &= f(\{a,c\},\ \{c\},\ \{a,c,d\}) \\ &= (\{a,c\},\ \{c\},\ \{a,c,d\}) \\ &= f^2(\emptyset,\ \emptyset,\ \emptyset) \end{aligned}$$

よって$(FIRST(A),FIRST(B),FIRST(C)) = (\{a,c\},\ \{c\},\ \{a,c,d\})$

レポート課題

- 教科書 p.84 Exercise 3.6