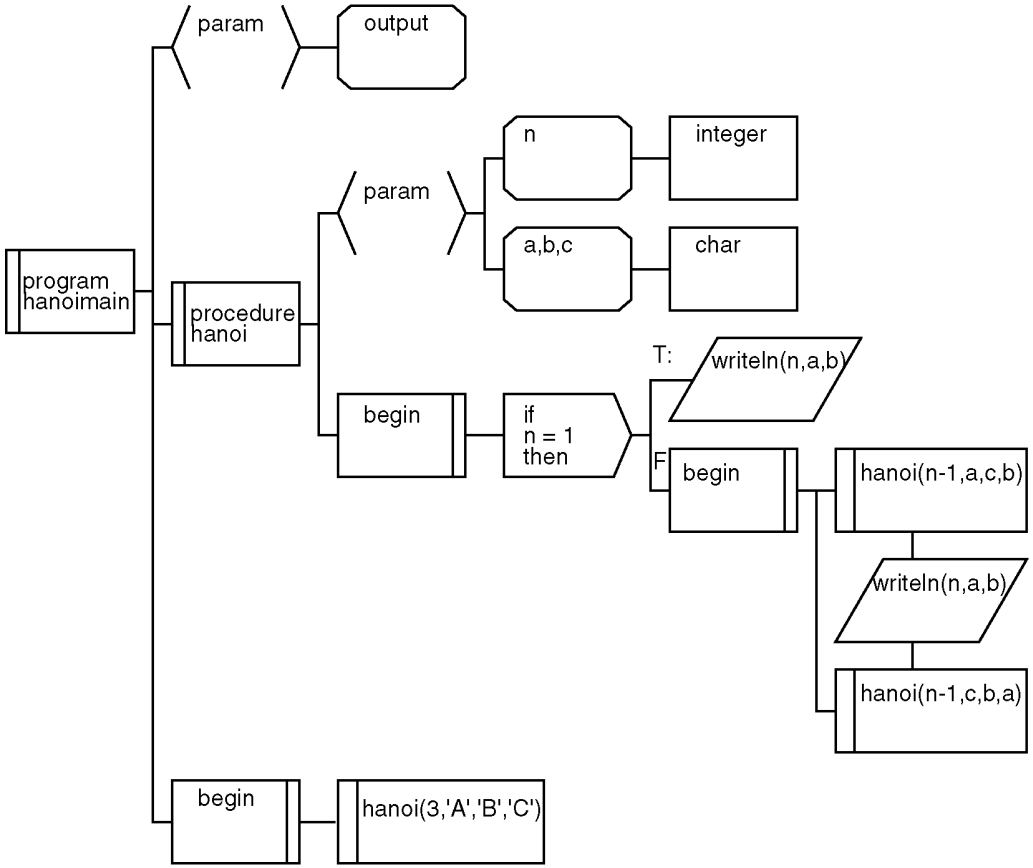
属性順位グラフ文法による プログラム図の構文解析

日本大学大学院
総合基礎科学研究科
地球情報数理科学専攻
類瀬健二

対象



プログラム図（Nichart）

目次

1. はじめに

HIerarchical flowCHART description language

- 1978年　夜久、二木
- 階層型流れ図言語
- 繰り返し記号を
  はじめて導入
- 疑似木構造グラフ

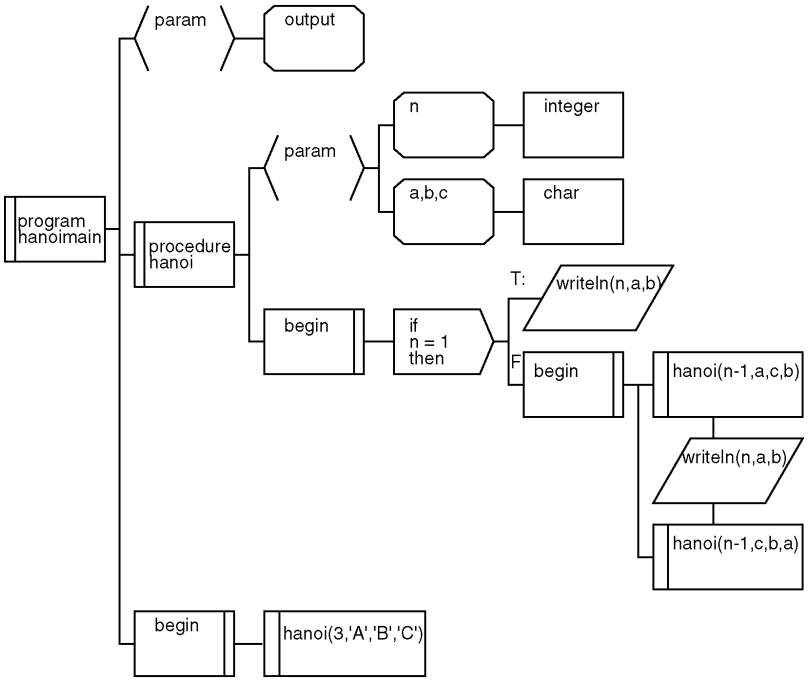
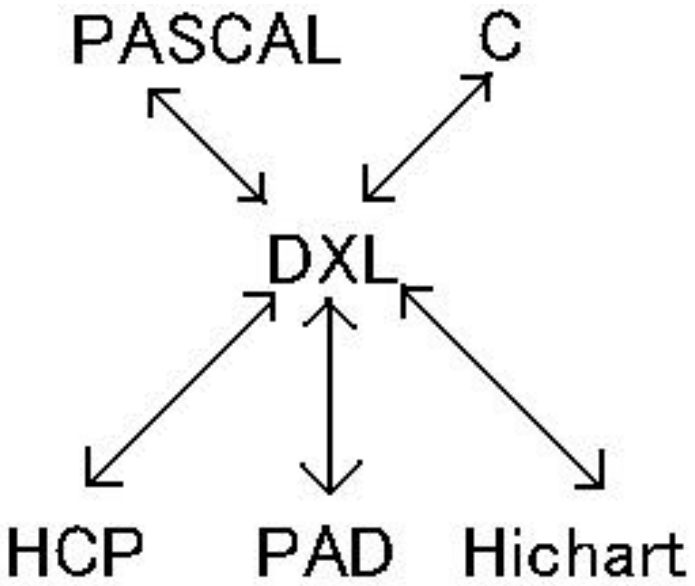


Diagram eXchange Language for tree structured charts

- 木構造図データ交換言語
- データの再利用や流通を促進
- JIS X 0130 (1995)



背景

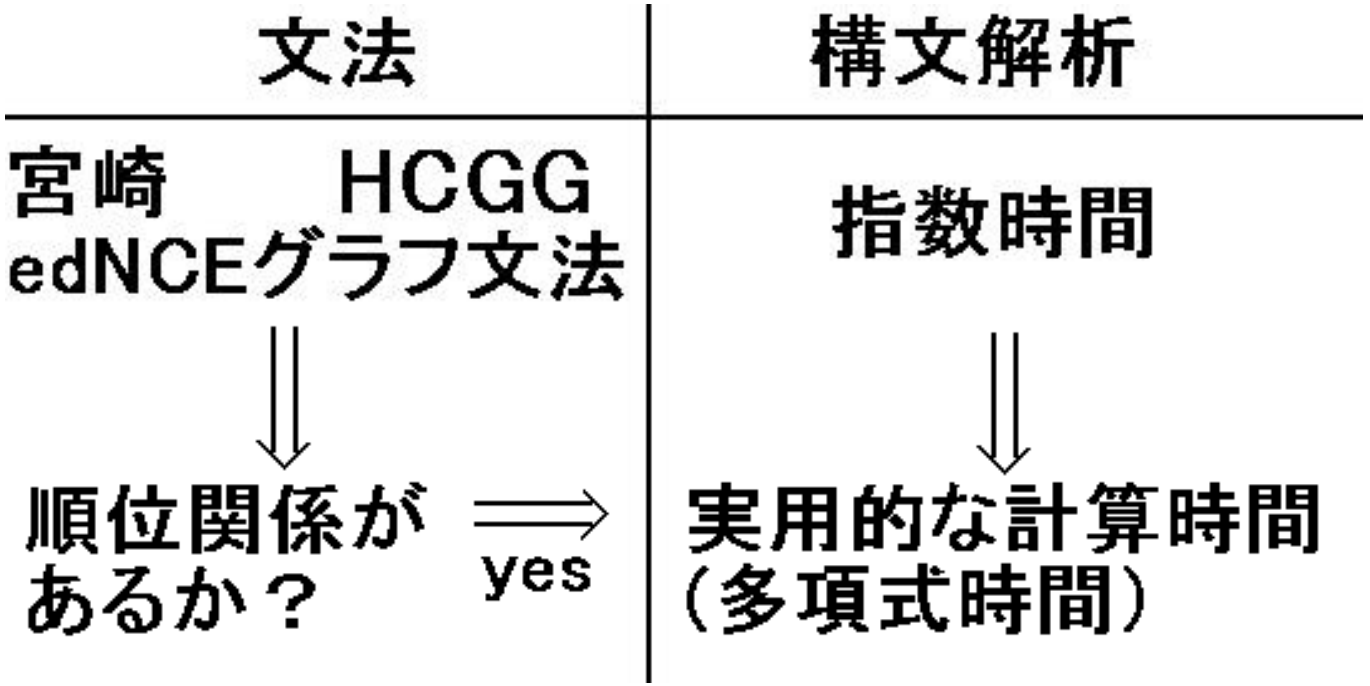
木構造図

- SPD
  日本電気株式会社
  (1975頃)
- Hichart
  夜久，二木(1978)
- PAD
  日立中央研究所(1979)
- 汎用木構造図記述言語
  DXL JIS X 0130(1995)

グラフ文法

- 文脈自由グラフ文法
  Vigna(1978)
- edNCEグラフ文法
  Janssens-Rozenberg(1982)
- DXL対応Hichartのための
  属性グラフ文法
  大井ら（Vigna-CFGG）
  COMP研（996）
- DXL対応Hichartのための
  属性グラフ文法 **HCGG**
  宮崎ら(edNCEグラフ文法)
  COMP研(2000)

動機


文法
構文解析
宮崎　　HCGG
edNCEグラフ文法
指数時間
順位関係が
あるか？
yes
実用的な計算時間
(多項式時間)

目的

(1)HCGGと等価な順位グラフ文法があるか確かめる

(2)順位関係を用いた構文解析アルゴリズムの構築および計算量の評価

(3)Parserの開発

結果

(1) (a)Hichart対応順位グラフ文法を構成

(b)生成規則69個，属性規則728個，

順位関係606個

(2)構文解析アルゴリズムの構築及び
計算量の評価(O(m))

(3)Parserの開発(約7000行)

2. 準備

- edNCEグラフ文法
- 属性edNCEグラフ文法
- 順位グラフ文法
- Hichart
- DXL
- DXL対応Hichartのための
  属性edNCEグラフ文法

2.1 edNCEグラフ文法[7]

- Σ :頂点アルファベット
- Γ :辺アルファベット

Σ，Γ上のグラフ :H=(V,E,ψ)

(1) V:頂点の有限集合
(2) E:辺のラベルのアルファベット　(V，Γ，V)
(3) ψ:頂点をラベル付けする関数　(ψ(V)=Σ)

edNCEグラフ文法：

6項組 GG＝（Σ，Δ，Γ，Ω，R，S）

- Σ ：頂点アルファベット
- Δ ：終端頂点アルファベット
- Γ ：辺アルファベット
- Ω ：終端辺アルファベット(Ω⊆Γ)
- R ：生成規則の有限集合
- S ：初期状態 (S∈Σ - Δ)

生成規則

$X \rightarrow (D,C)$

X :非終端な頂点アルファベット

(D,C): Σ, Γ 上の埋め込みを持つグラフ

$D \in GR_{\Sigma,\Gamma}$:Σ とΓ 上のグラフ

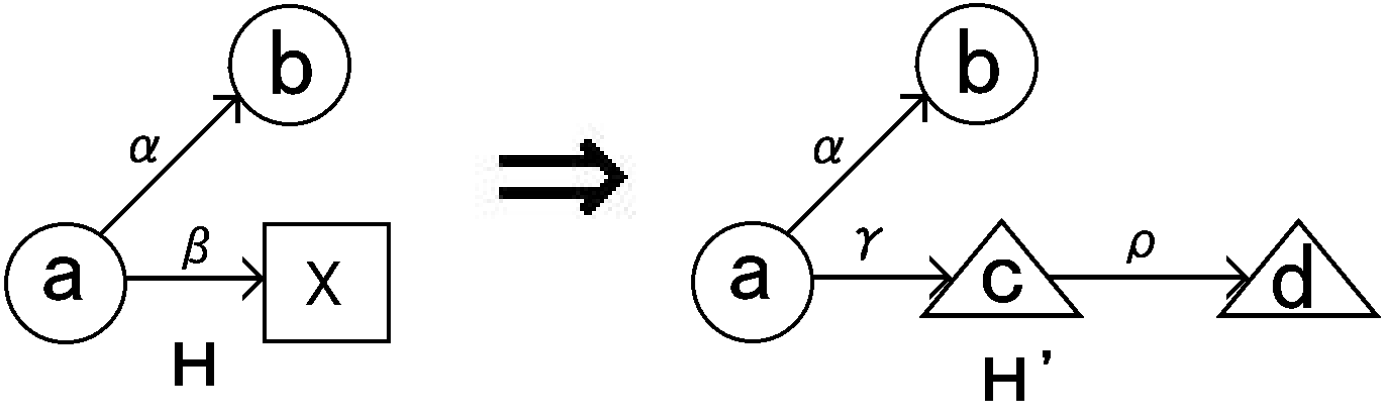
$C \subset \Sigma \times \Gamma \times \Gamma \times V_D \times \{in,out\}$:(D,C)の接続関係

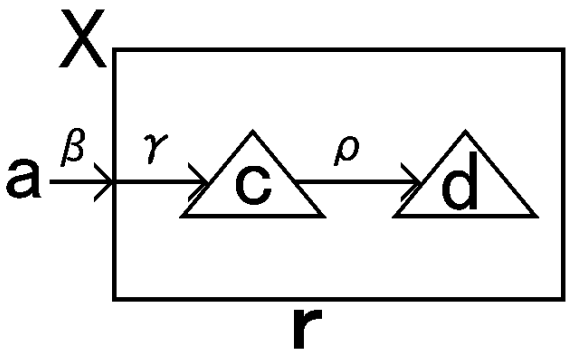
$\delta \in \Sigma$, $\beta, \gamma \in \Gamma$, $x \in V_D$, $d \in \{in,out\}$

Cのそれぞれの要素 $(\delta, \beta, \gamma, x, d)$:

(D,C)の接続命令

導出の例





生成規則

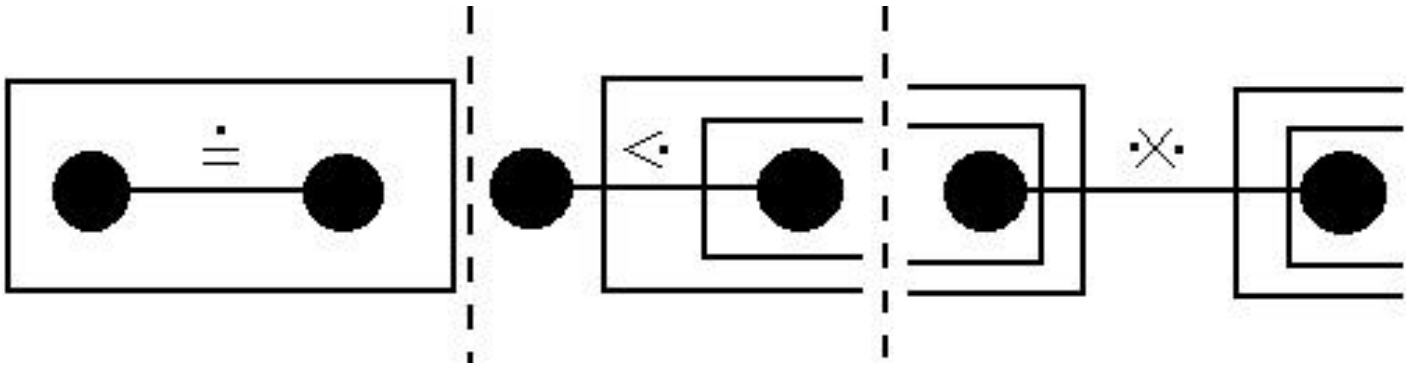
2.2 属性edNCEグラフ文法[8]

AGG=＜GG,A,F＞の 3項組

- GG :基底edNCEグラフ文法
- A :属性集合
- F :意味規則集合

2.3 順位グラフ文法[10]

定義2.10[10] $\bowtie \in \{\doteq, <\cdot, \cdot>, \cdot\times\cdot\}$は2頂点間の順位関係である.
そのラベル間の順位関係は$R_{\bowtie}$であり, $\bowtie$はすべての$lab_G(\mathrm{v},\mathrm{w})$の集合である. □



順位グラフ文法は次の6項組である.

$GG_{\bowtie} = (\Sigma, \Delta, \Gamma_{\bowtie}, \Omega_{\bowtie}, R, S)$

- $\Gamma_{\bowtie} \stackrel{def}{=} \Gamma \times \{\doteq, \lessdot, \gtrdot, \times\}$
- $\Omega_{\bowtie} \stackrel{def}{=} \Omega \times \{\doteq, \lessdot, \gtrdot, \times\}$

  拡張辺ラベルアルファベット(extended edge label alphabet)

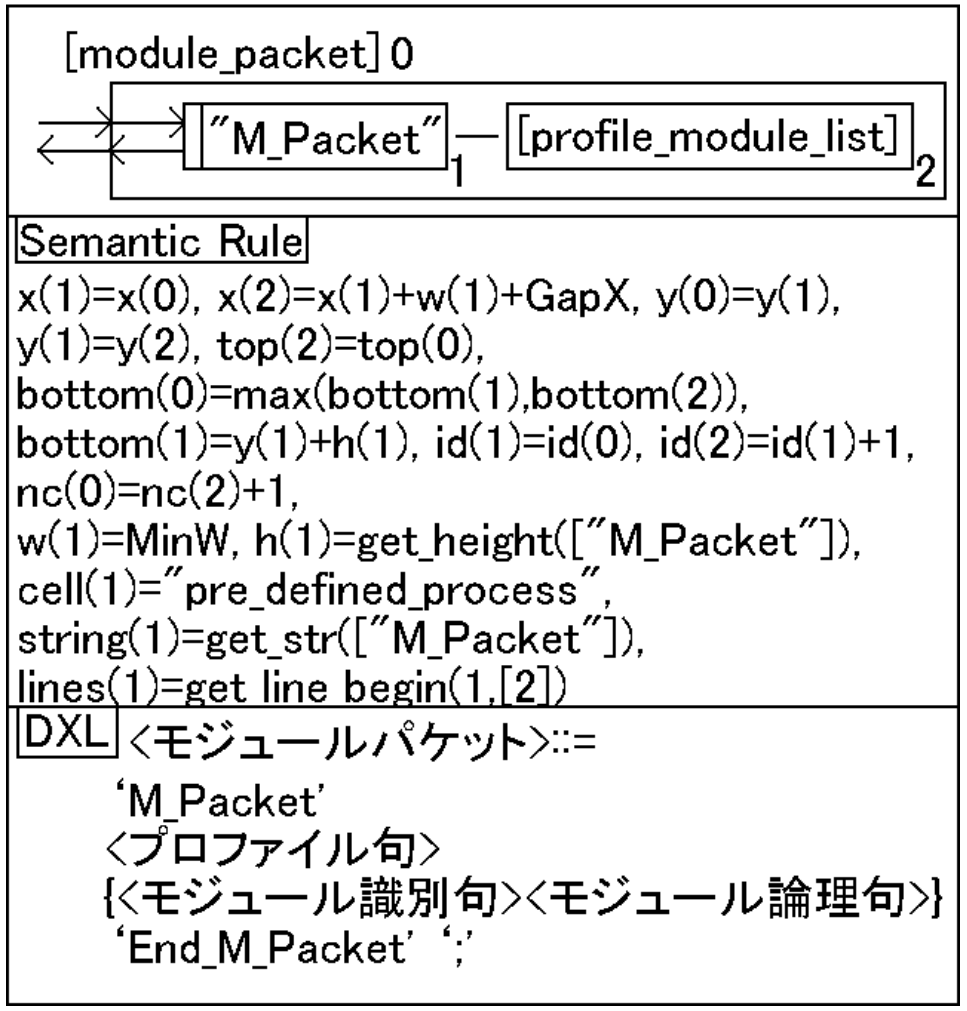
2.4 HIerarchical flowCHART description language[1]

2.5 Diagram eXchange Language for tree structured charts[6]

2.6 HCGG[8]

HCGG：
DXL対応Hichartの
ための属性edNCE
グラフ文法

- 生成規則67個
- 属性規則数716個



3. HCGGに対する順位グラフ文法

- Hichart対応順位グラフ文法

3.1 Hichart対応順位グラフ文法

- Hichart対応グラフ文法[8]を修正して順位関係を組み込み，Hichart対応順位グラフ文法（HCPGG）を定式化

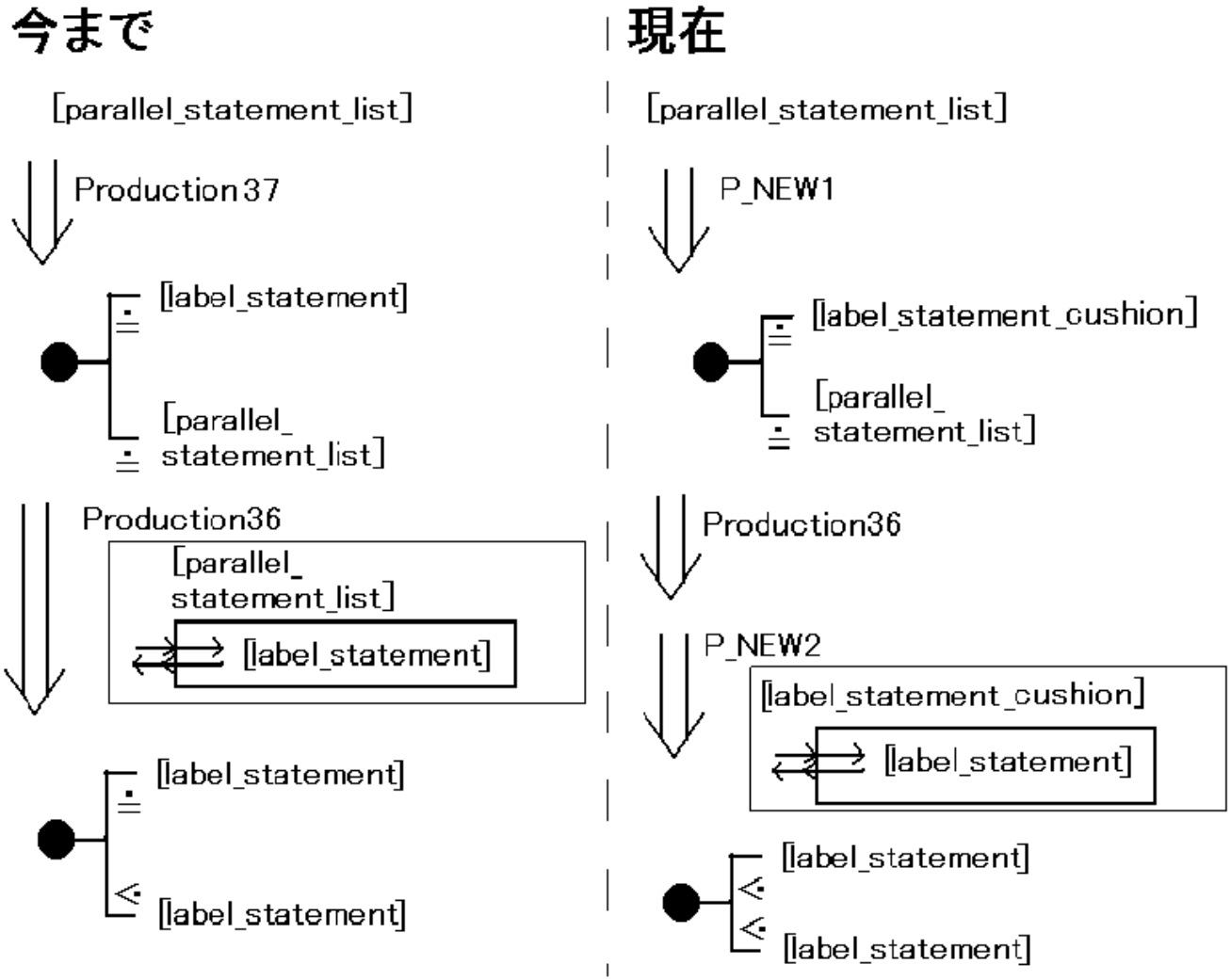
HCPGG:

DXL対応Hichartのための

属性edNCEグラフ文法

- 生成規則69個
- 意味規則728個

- 生成規則の修正，追加


今まで
[parallel_statement_list]
Production37
[label_statement]
[parallel_
statement_list]
Production36
[parallel_
statement_list]
[label_statement]
[label_statement]
[label_statement]
現在
[parallel_statement_list]
P_NEW1
[label_statement_cushion]
[parallel_
statement_list]
Production36
P_NEW2
[label_statement_cushion]
[label_statement]
[label_statement]
[label_statement]

定理３．２

L(HCGG)を生成する順位グラフ文法が存在する

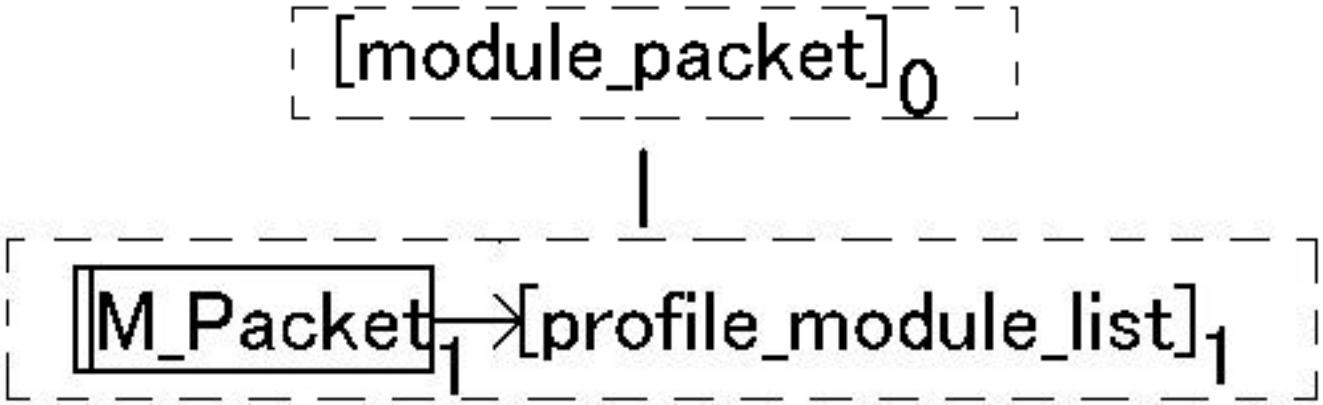
証明）HCPGGを考える

HCPGGの頂点アルファベット間において、
順位関係を矛盾なく定義できる

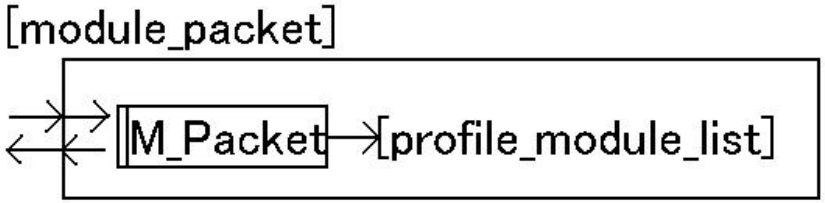
- 頂点アルファベット間の順位関係表

http://www.am.chs.nihon-u.ac.jp/~yaku/archive
/thesis/6199M13/ronbun/paper.pdf

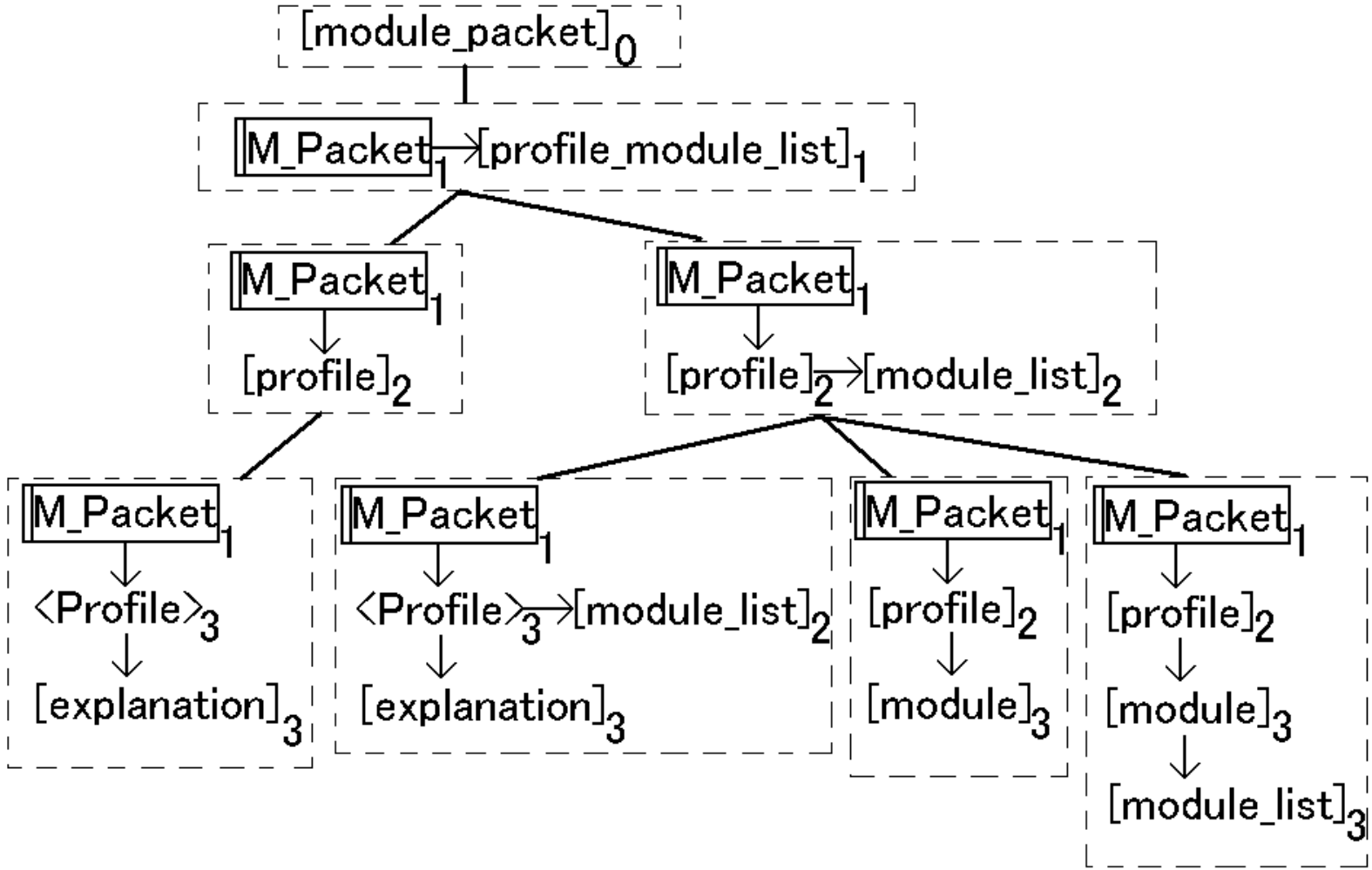
- 頂点間の順位関係の計算

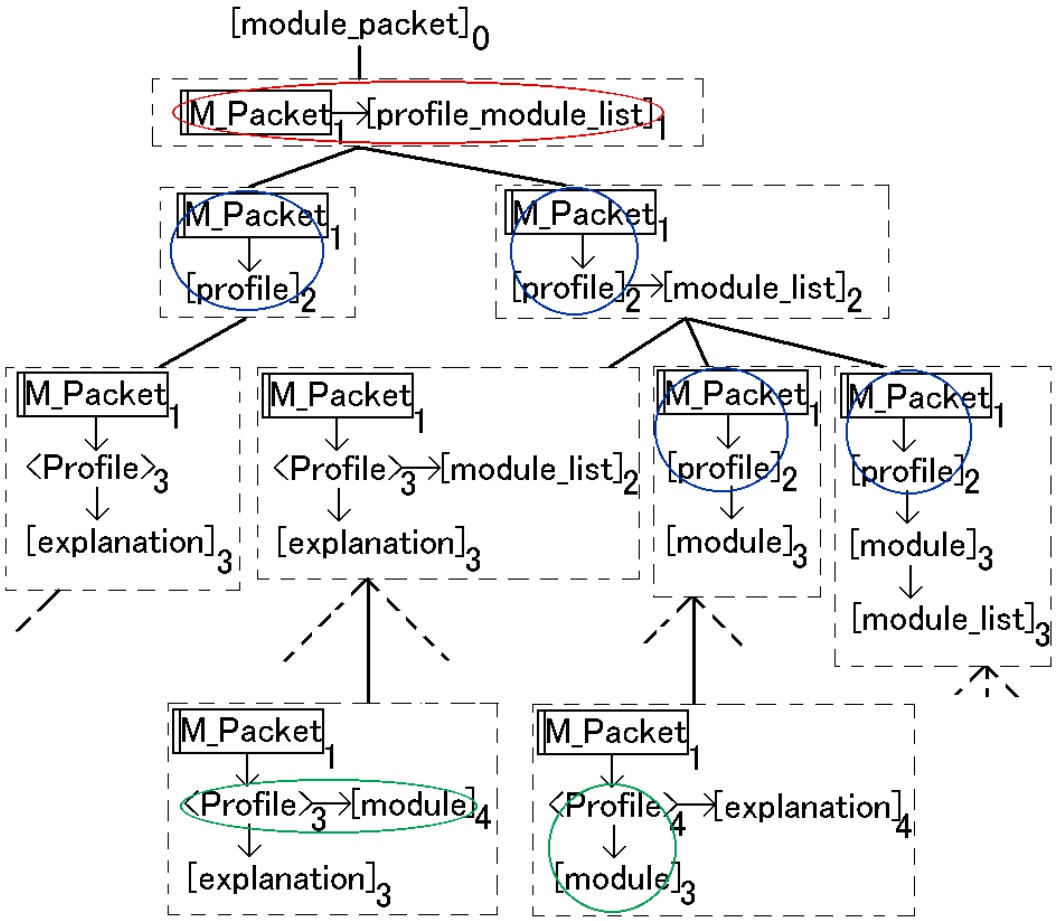


生成規則



HCPGGのHasse diagram(一部)





- $(\boxed{\text{M\_Packet}}, \#, \phi, [\text{profile\_module\_list}]) \in R_{=}$
- $(\boxed{\text{M\_Packet}}, \#, \phi, [\text{profile}]) \in R_{\swarrow}$
- $(\langle\text{Profile}\rangle, \#, \phi, [\text{module}]) \in R_{\swarrow}$

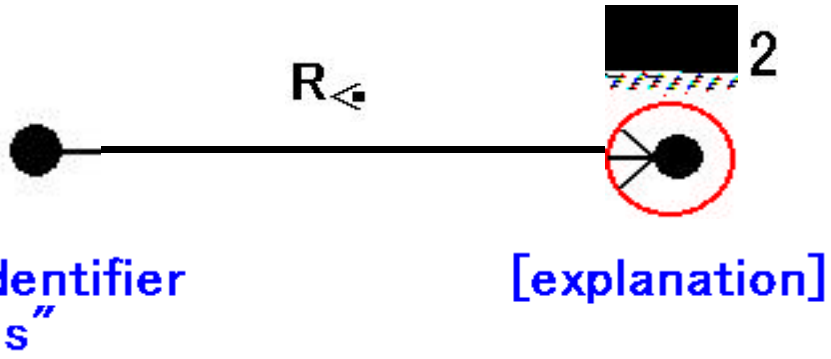
- 順位関係606個(一部)

	[module_list]	"Profile"	[explanation]	[module]	"Identifier is"
[module _profile]					
"M_Packet"		$R_{\leqslant}$			
[profile _module _list]					
[profile]	$R_{=}$			$R_{\leqslant}$	$R_{\leqslant}$
[module _list]					
"Profile"	$R_{>}$		$R_{=}$	$R_{\times}$	$R_{\times}$
[explanation]					
[module]	$R_{=}$			$R_{\leqslant}$	$R_{\leqslant}$
"Identifier _is"	$R_{>}$		$R_{\leqslant}$	$R_{\times}$	$R_{\times}$
[explanation _module _algorithm]					

□

順位関係の応用

$lab_G(1,2)=(\text{"Identifier\_is"},\#,\phi,\text{[explanation]})$



	[module_list]	"Profile"	[explanation]	[module]	"Identifier is"
"Profile"	$R_{>}$		$R_{=}$	$R_{\times}$	$R_{\times}$
[explanation]					
[module]	$R_{=}$			$R_{<}$	$R_{<}$
"Identifier _is"	$R_{>}$		$R_{<}$	$R_{\times}$	$R_{\times}$
[explanation _module _algorithm]					

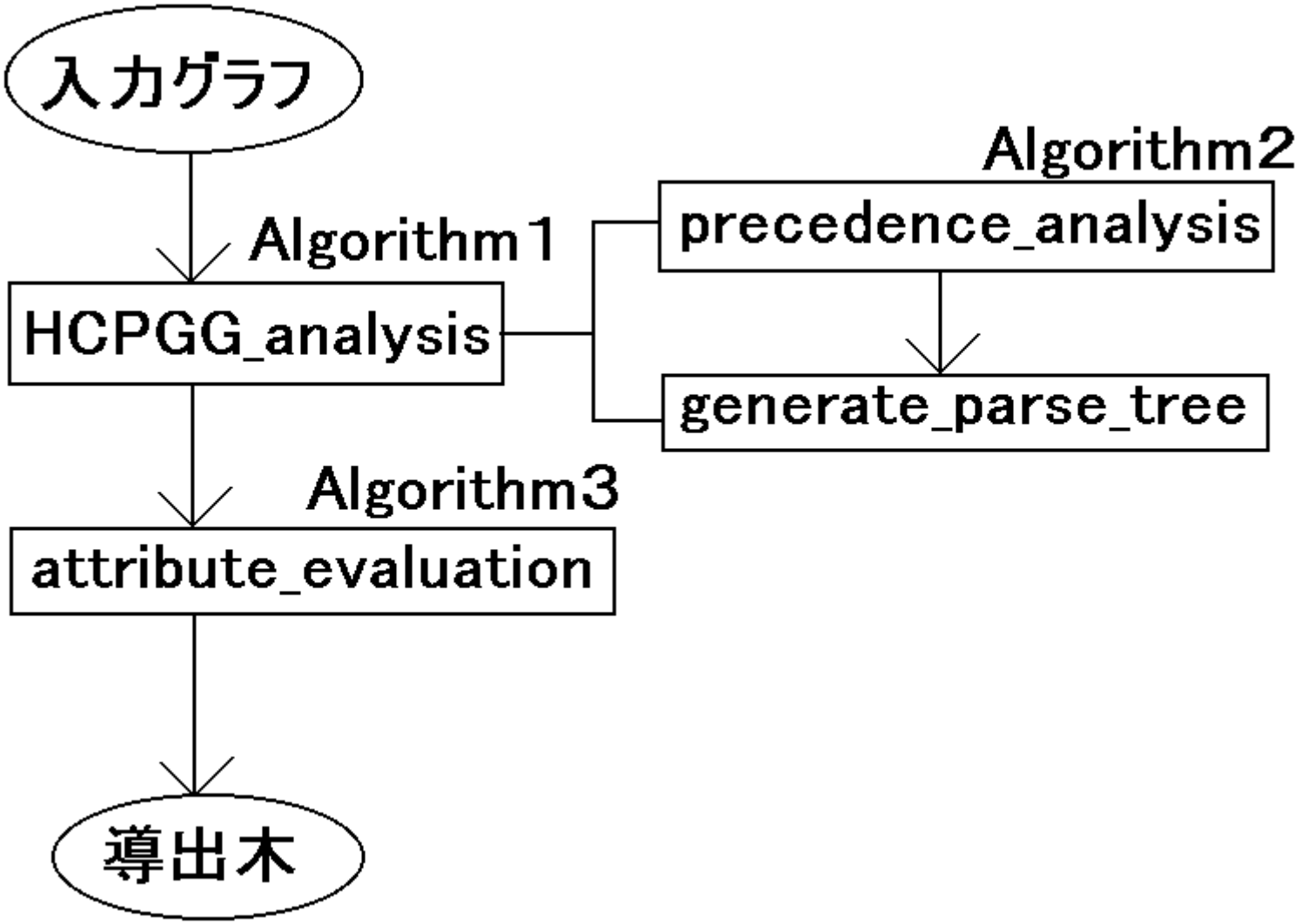
4. HCPGGに対する構文解析

- Hichart対応順位グラフ文法に対しての構文解析アルゴリズムの構築
- HCPGGに対する計算量の評価

4.1　HCPGGに対する構文解析アルゴリズム

- Hichart対応順位グラフ文法に対しての構文解析アルゴリズムを構築
  （Kaul[10]のアルゴリズムと属性評価）

全体


入力グラフ
Algorithm1
HCPGG_analysis
Algorithm2
precedence_analysis
generate_parse_tree
Algorithm3
attribute_evaluation
導出木

Algorithm 1:

HCPGG_analysis

(Kaul[10])

入力グラフを与え，

- **precedence_analysis**

構文解析可能かを調べる

(algorithm 2)

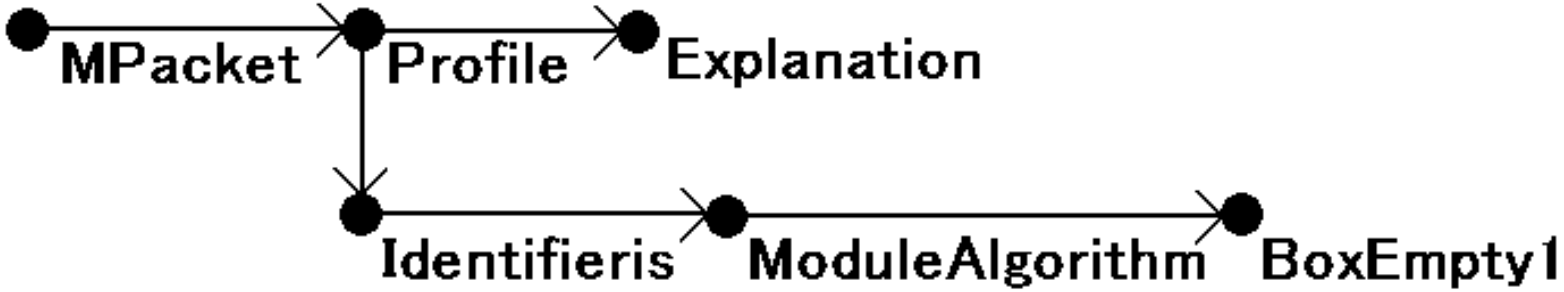
- **generate_parse_tree**

構文木の出力

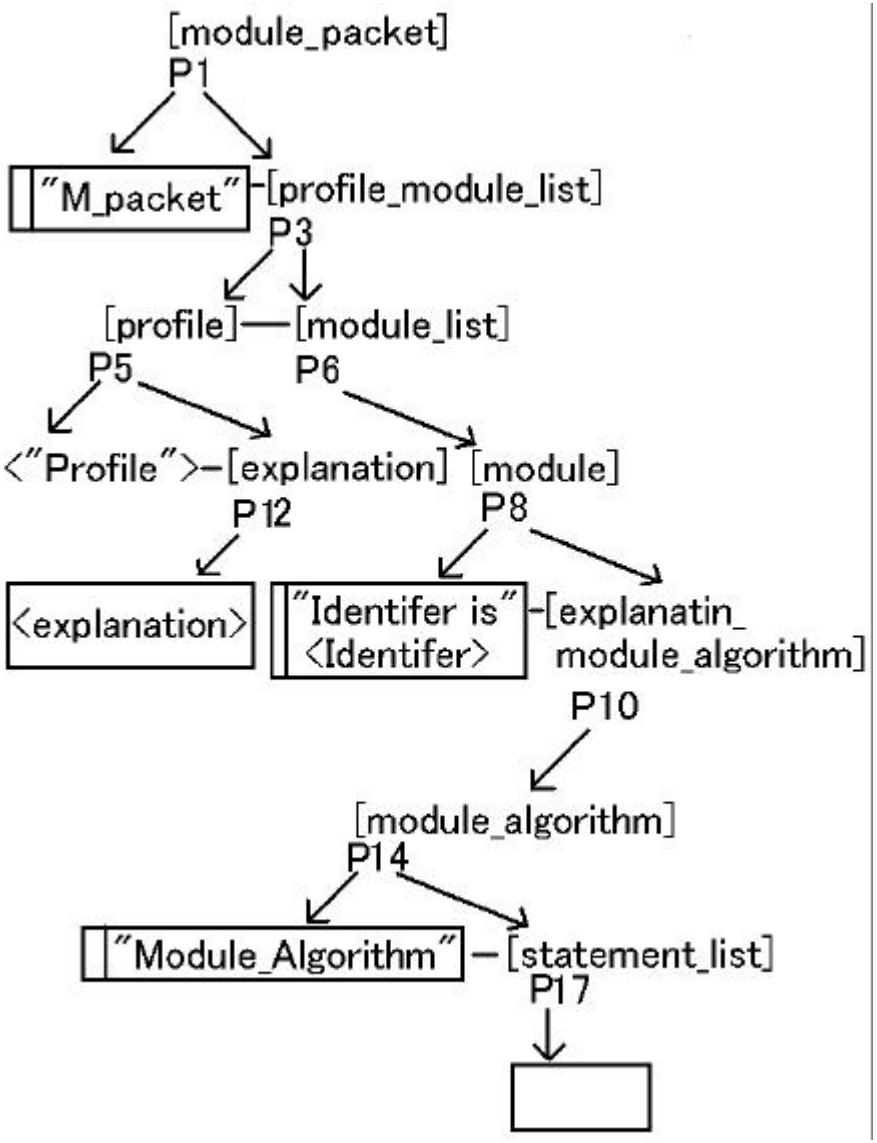
Algorithm 1

```
HCPGG_analysis(Graph G){
  while(G!=start graph){
    G=prcedence_analysis(G);
      (algorithm 2)
  }
  generate_parse_tree(T);
}
```

Algorithm1の入力（例）



Algorithm1の出力(例)



Algorithm 2 (Kaul[10])

precedence_analysis

入力 :グラフ

shift_order_list
順序ハンドルの探索
頂点の順序リスト

- find_production
  生成規則の発見
- replace_graph
  グラフの書き換え
- rewriting_order_list
  順序リストの書き換え

出力 :置き換えたグラフ

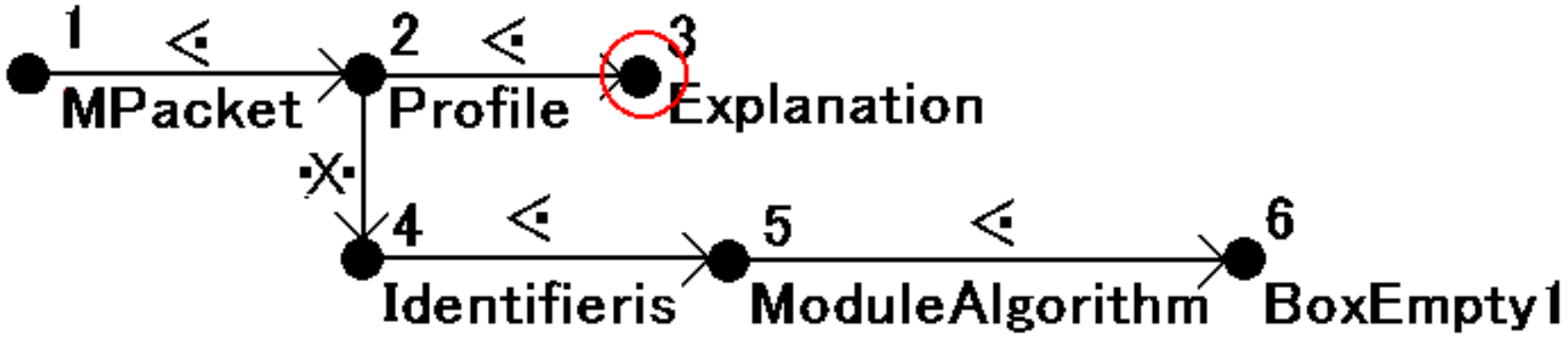
Algorithm 2

```
Graph precedence_analysis
    (Graph G){
    K=shift_order_list(G);          (1)
    p=find_production(K);           (2)
    G=replace_graph(G,K,p);         (3)
    K=rewriting_order_list(K,P);(4)
}
```

例　precedence_analysisの動作

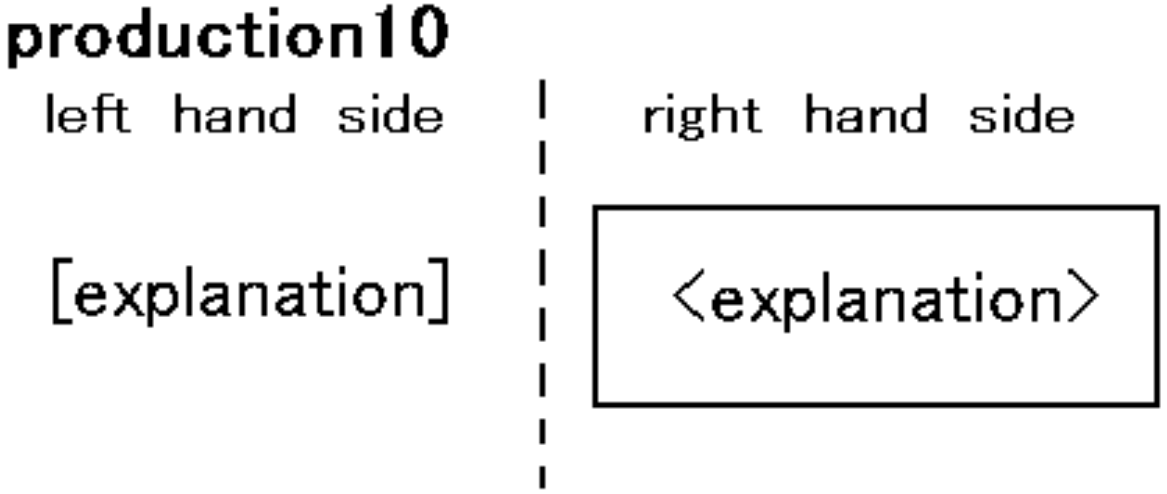
(1) K=shift_order_list(G);

　　Ｓｈｉｆｔの動作を行なう Ｋaul[10])



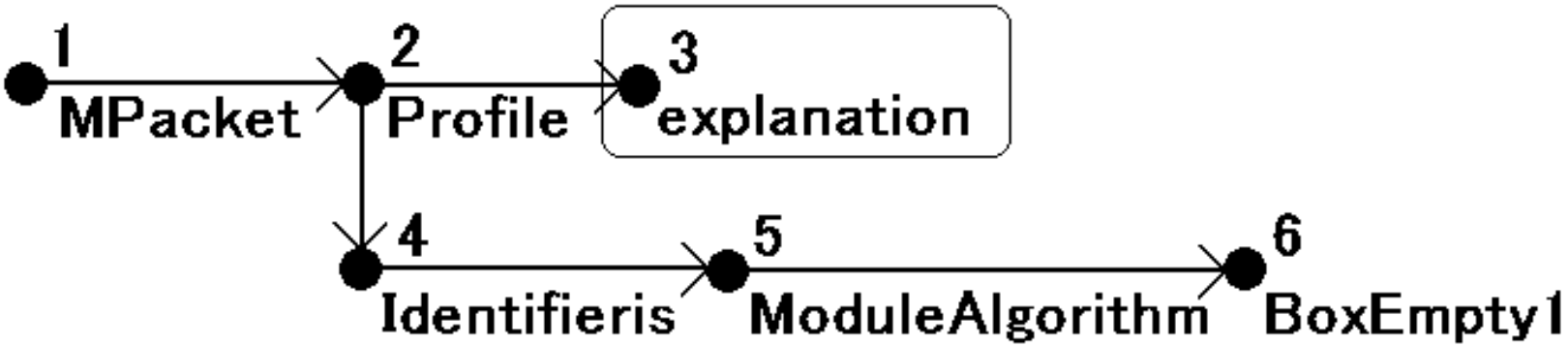
(2) p=find_production(K);

・順位ハンドルと一致する生成規則の
right hand sideの探索（Kaul[10]）



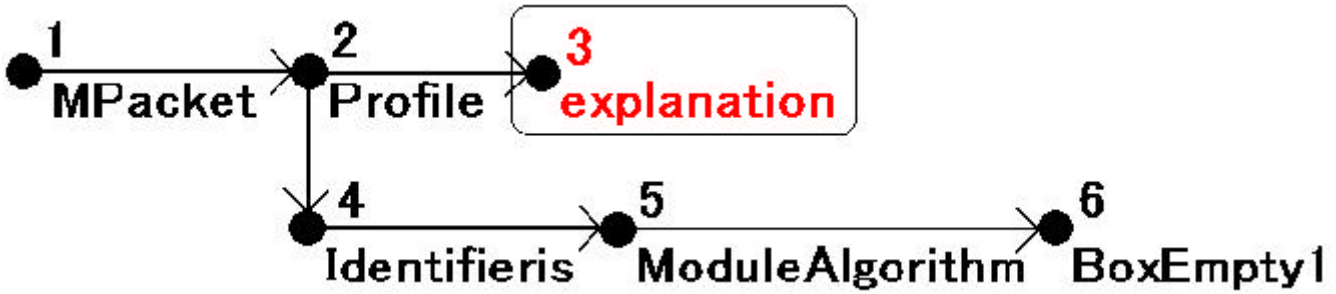
(3) **G=replace_graph(G,K,P);**

グラフの置き換えを行なう(Kaul[10])



⑷ K=rewriting_order_list(K,p);

順序リストの書き換え(Kaul[10])

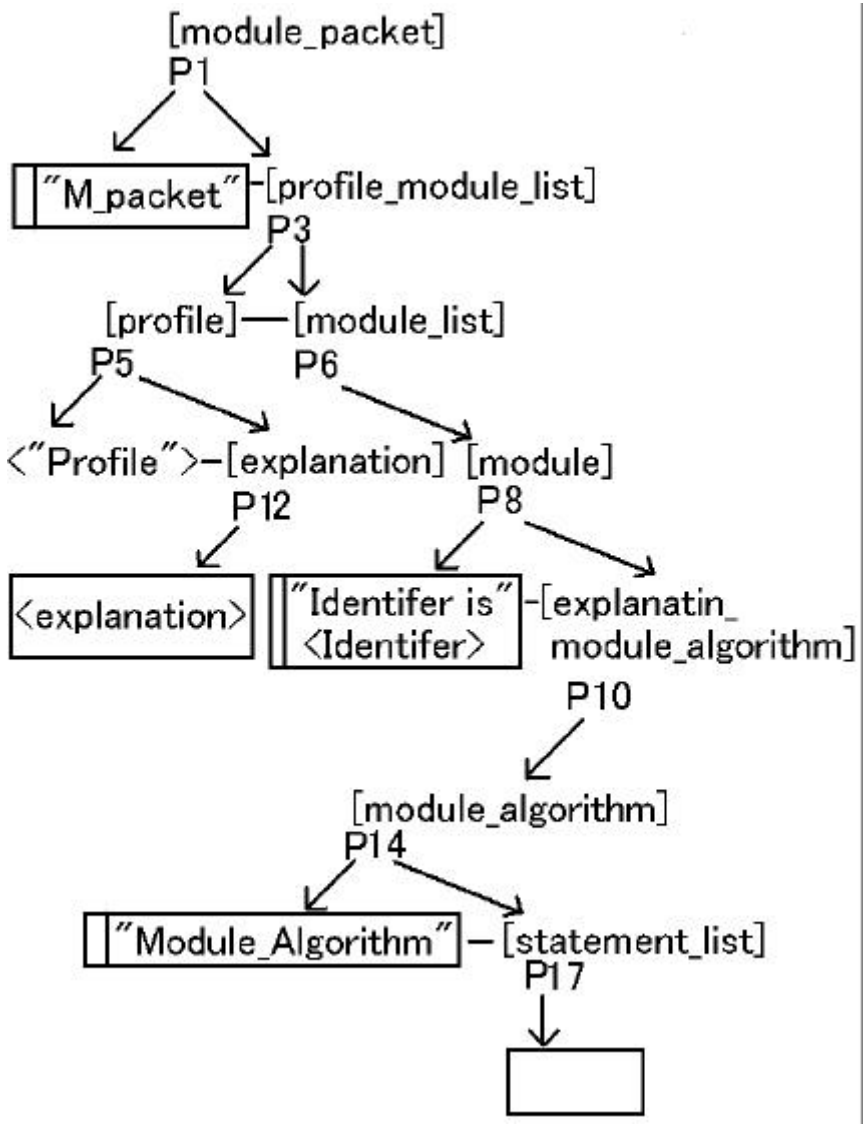


K=[1,2,3]
(3の頂点アルファベット= explanation)

□

Algorithm1で
作成した構文木

例



attribute_evaluation

- 属性評価を行なうためのAlgorithm

Algorithm3
attribute_evaluation{
attribute_parse_tree; (属性計算を行なう順列)　(1)
attribute_calculate;　(属性計算を行なう)　(2)
}
}

attribute_parse_tree;

⑴属性計算を行なう順列を求める

例

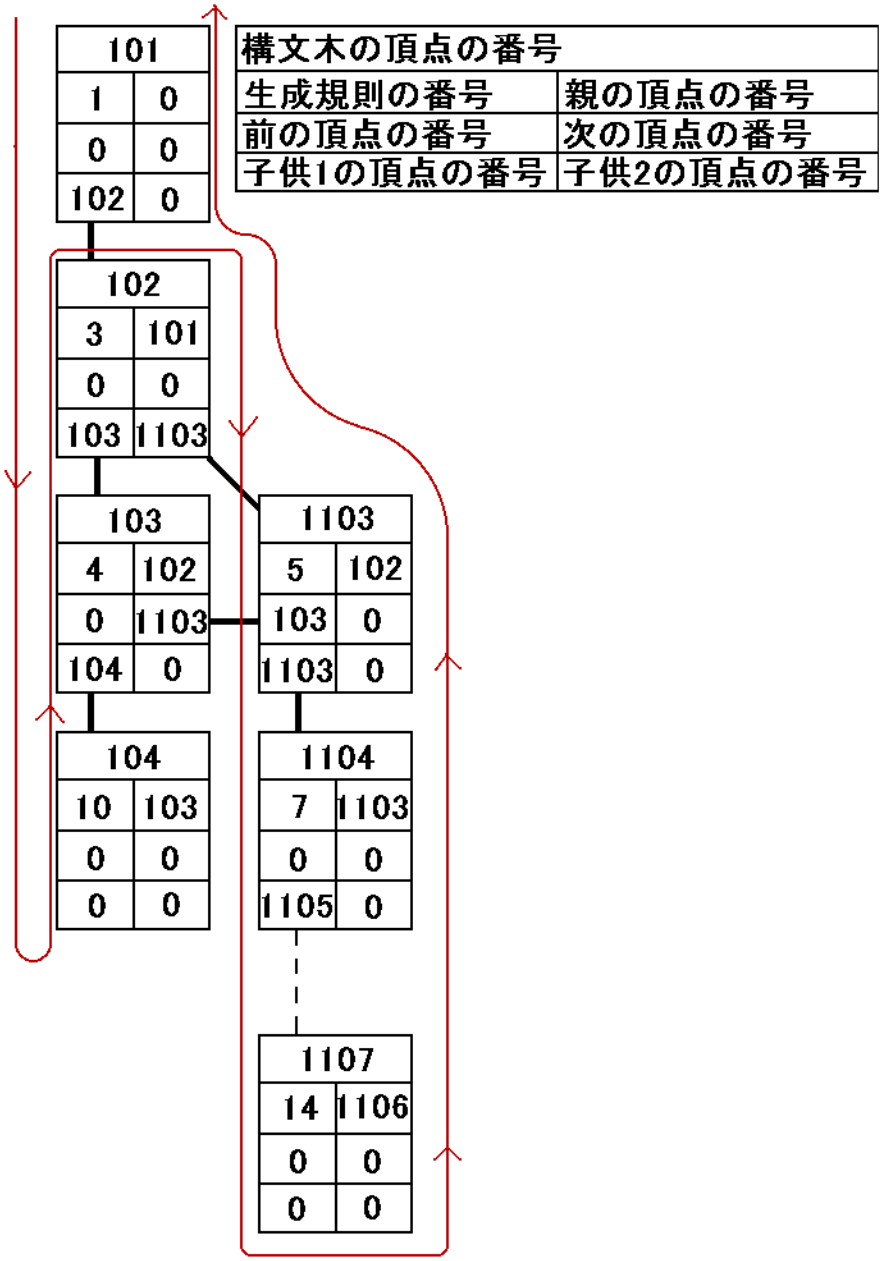
1 :頂点番号101の継承

2 :頂点番号102の継承

……

17 :頂点番号102の合成

18 :頂点番号101の合成



(2) Attribute_calcurate;

属性評価を行なう

- 属性の評価

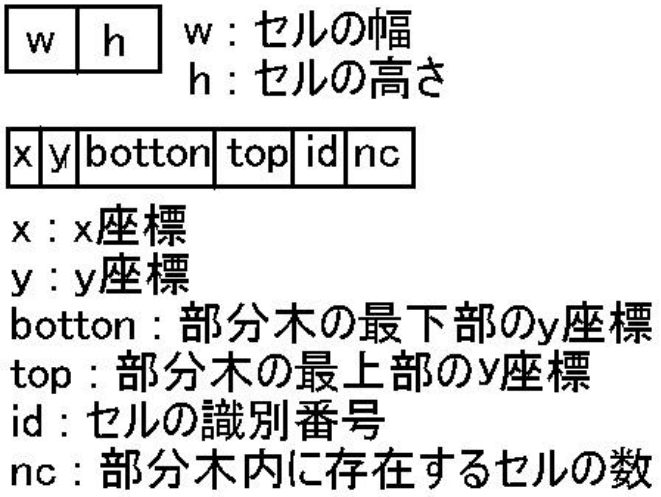
(1)継承属性

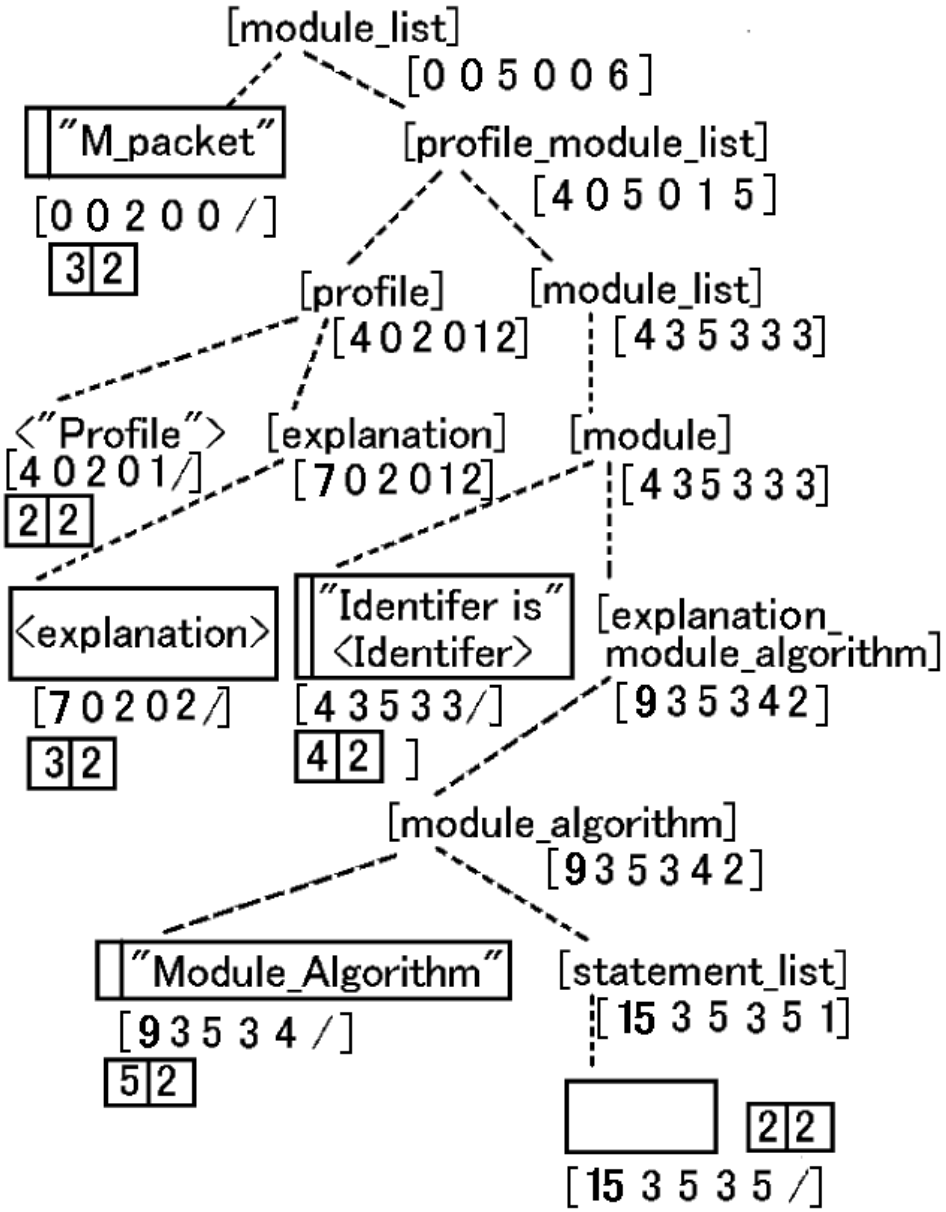
(2)合成属性

によって行なわれる

例　Algorithm 3の実行例

構文木に対して属性評価を行なう
(大井ら[5])





Algorithm3の出力・図1

Algorithm 3の応用 •描画

- HCPGG_parserと属性評価（大井ら）の結果を受けてDiagramの描画を行なう

例

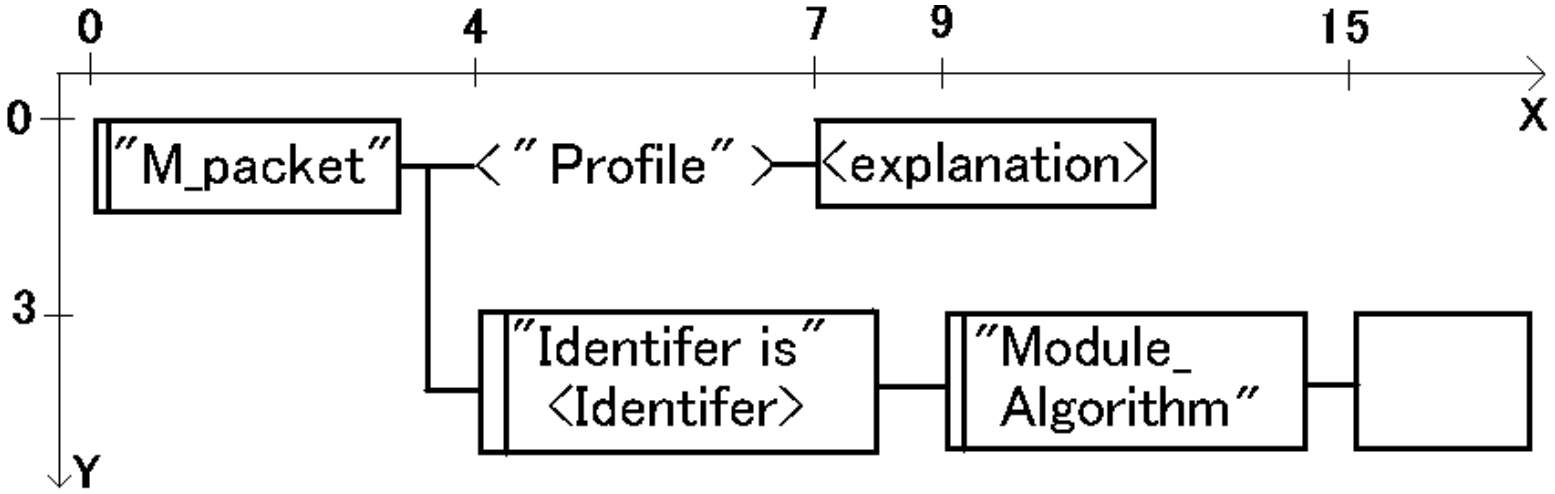


図1の描画

4.2 HCPGGによる構文解析アルゴリズムの計算時間

定理4.1

HCPGG_analysisは線形時間（O(m)）で実行可能である

証明. 略（Kasamiの一般形：$O(n^2)$） □

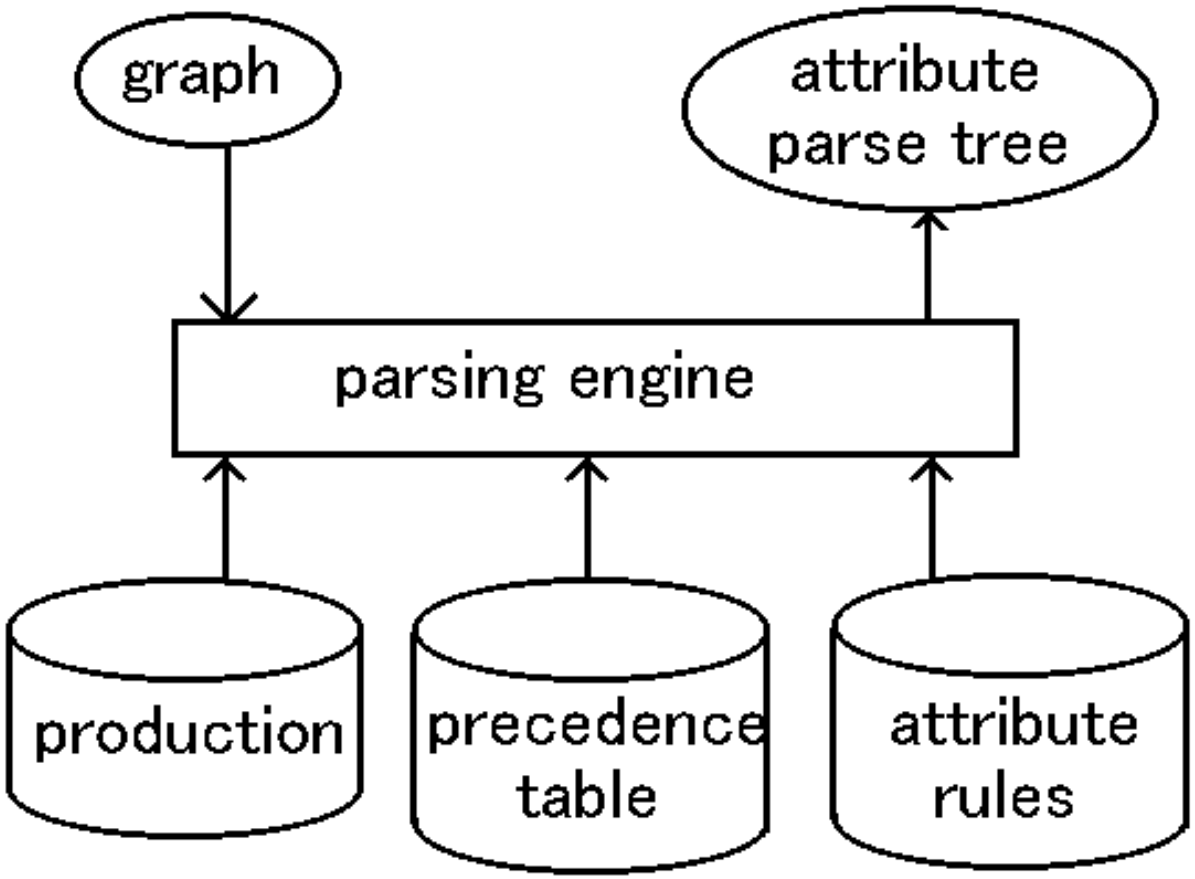
定理4.2

attribute_evaluationは線形時間（O(n)）で実行可能である

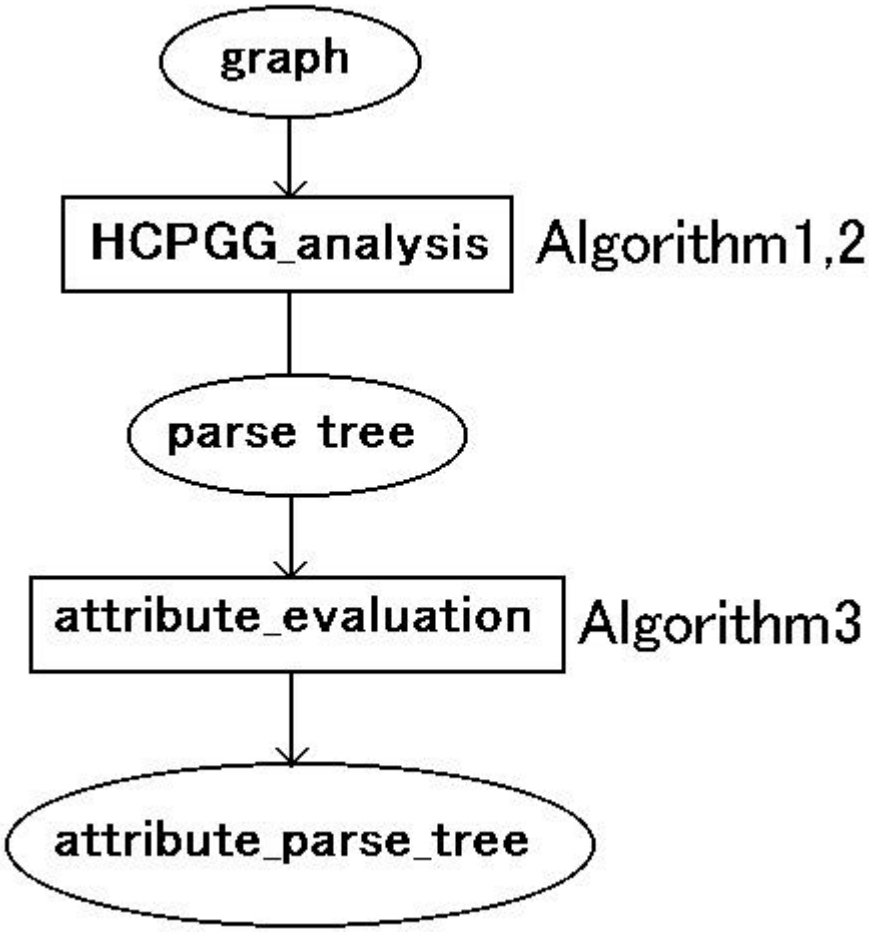
証明. 略 □

5. HCPGG Parser

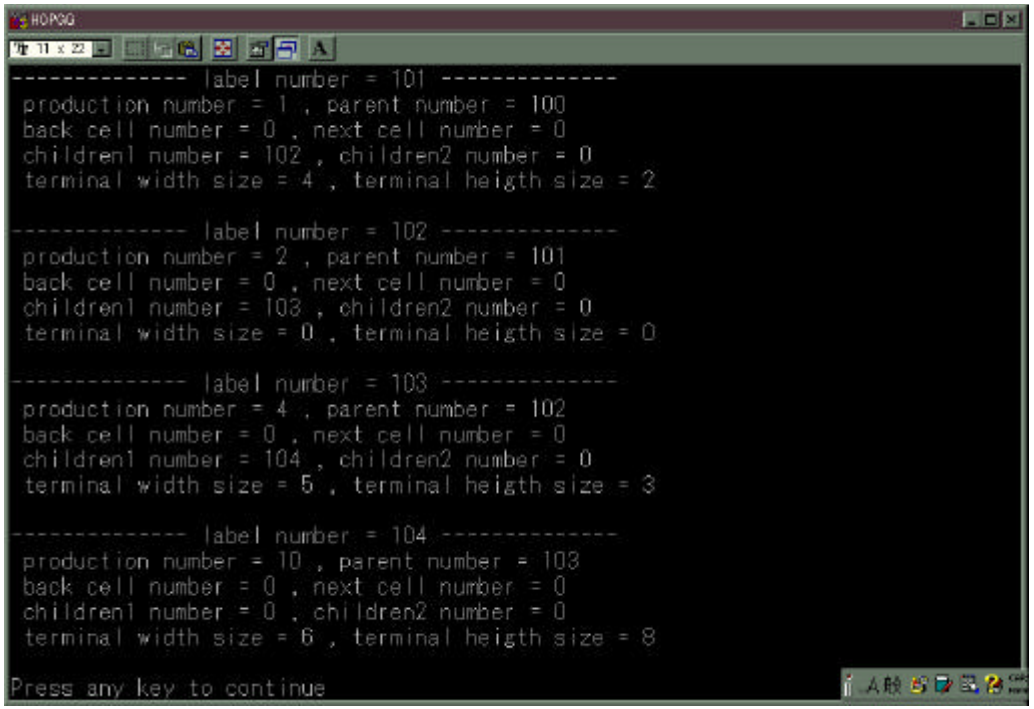
内部構造

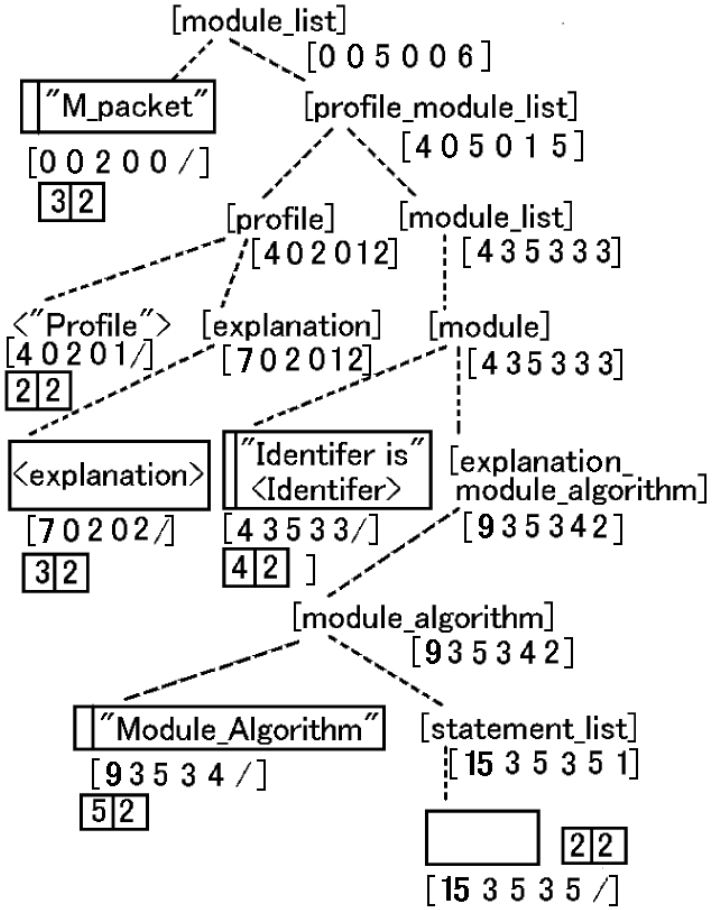

graph
attribute
parse tree
parsing engine
production
precedence
table
attribute
rules

制御の流れ


graph
HCPGG_analysis
Algorithm1,2
parse tree
attribute_evaluation
Algorithm3
attribute_parse_tree

入出力


HOPGO
11 x 22
label number = 101
production number = 1 , parent number = 100
back cell number = 0 , next cell number = 0
children1 number = 102 , children2 number = 0
terminal width size = 4 , terminal heigth size = 2
label number = 102
production number = 2 , parent number = 101
back cell number = 0 , next cell number = 0
children1 number = 103 , children2 number = 0
terminal width size = 0 , terminal heigth size = 0
label number = 103
production number = 4 , parent number = 102
back cell number = 0 , next cell number = 0
children1 number = 104 , children2 number = 0
terminal width size = 5 , terminal heigth size = 3
label number = 104
production number = 10 , parent number = 103
back cell number = 0 , next cell number = 0
children1 number = 0 , children2 number = 0
terminal width size = 6 , terminal heigth size = 8
Press any key to continue


[module_list]
[0 0 5 0 0 6]
"M_packet"
[0 0 2 0 0 /]
3 2
[profile_module_list]
[4 0 5 0 1 5]
[profile]
[4 0 2 0 1 2]
[module_list]
[4 3 5 3 3 3]
<"Profile">
[4 0 2 0 1/]
2 2
[explanation]
[7 0 2 0 1 2]
[module]
[4 3 5 3 3 3]
<explanation>
[7 0 2 0 2/]
3 2
"Identifer is" <Identifer>
[4 3 5 3 3/]
4 2]
[explanation_ module_algorithm]
[9 3 5 3 4 2]
[module_algorithm]
[9 3 5 3 4 2]
"Module_Algorithm"
[9 3 5 3 4 /]
5 2
[statement_list]
[15 3 5 3 5 1]
2 2
[15 3 5 3 5 /]

6 まとめ

(1)HCGGと等価な順位グラフ文法の存在

(2)構文解析アルゴリズムの構築および
　属性評価を含む計算量の評価

(3) Parserの作成（約7000行）

今後

- Hichart処理システムの拡張