SYNTACTIC TABULAR FORM PROCESSING BY PRECEDENCE ATTRIBUTE GRAPH GRAMMARS

Tomokazu ARITA (Nihon Univ.)

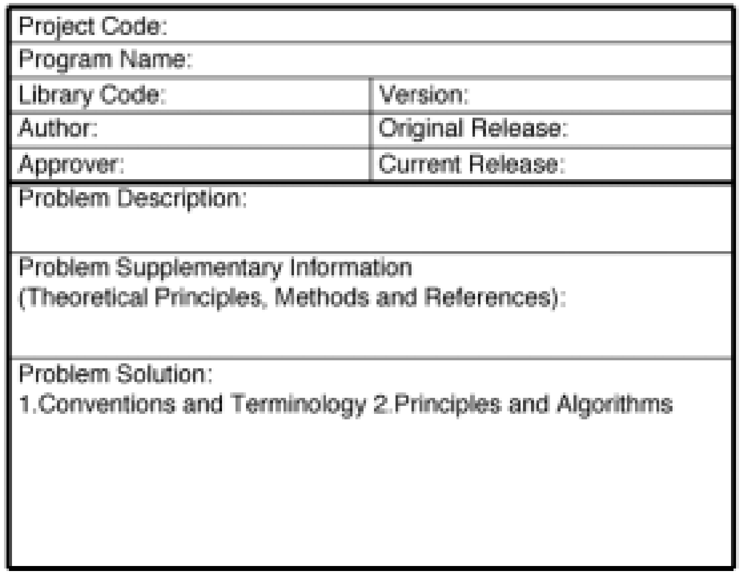
Kimio SUGITA (Tokai Univ.)

Kensei TSUCHIDA (Toyo Univ.)

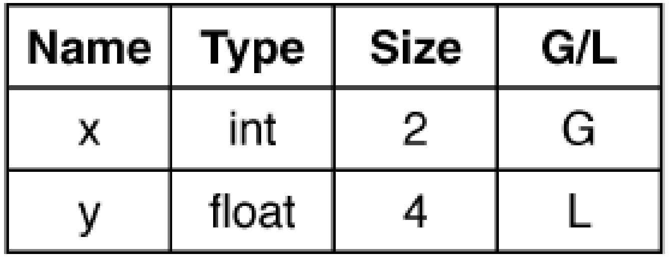
Takeo YAKU (Nihon Univ.)

Abstract

Target



Project Code:	
Program Name:	
Library Code:	Version:
Author:	Original Release:
Approver:	Current Release:
Problem Description:	
Problem Supplementary Information (Theoretical Principles, Methods and References):	
Problem Solution: 1.Conventions and Terminology 2.Principles and Algorithms	



Name	Type	Size	G/L
x	int	2	G
y	float	4	L

Goal

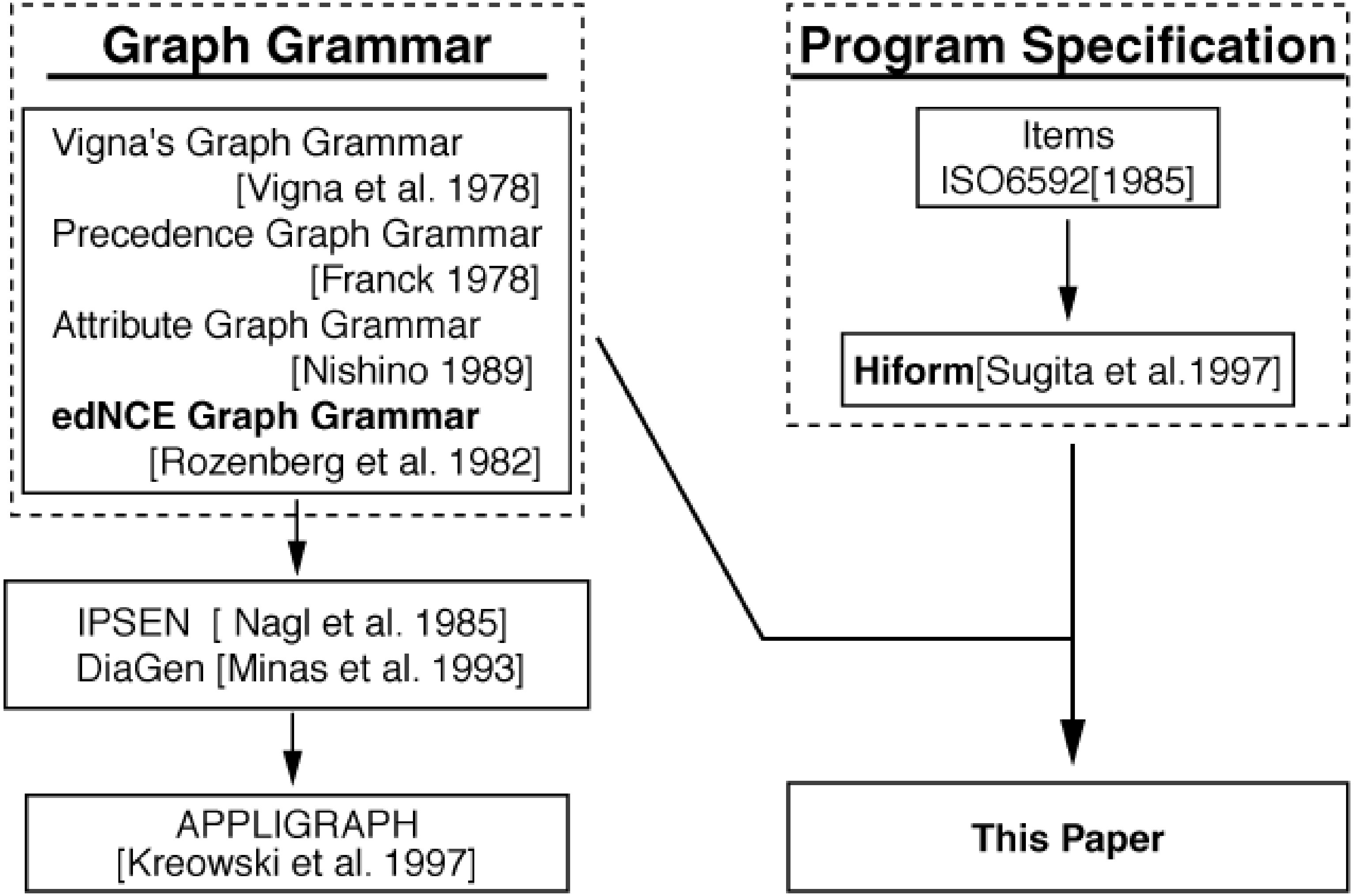
To construct development methods of a Tabular Form Processing System.

Contents

1. Introduction
2. Program specification language
3. Attribute graph grammar
4. An Attribute Graph Grammar for Hiform.
5. Parsing of Precedence Attribute NCE Graph Grammar
6. Conclusions

1. Introduction

Background



Graph Grammar

Vigna's Graph Grammar [Vigna et al. 1978]

Precedence Graph Grammar [Franck 1978]

Attribute Graph Grammar [Nishino 1989]

edNCE Graph Grammar [Rozenberg et al. 1982]

IPSEN [Nagl et al. 1985]

DiaGen [Minas et al. 1993]

APPLIGRAPH [Kreowski et al. 1997]

Program Specification

Items ISO6592[1985]

Hiform[Sugita et al.1997]

This Paper

Motivation

In mechanical documentation,
it is necessary to formally define tabular forms and the drawing conditions.

Purpose

- To propose a model for forming tabular forms efficiently.
- To construct an application of this model for tabular forms.
- To investigate properties of this application.
- To propose an analyzing method of this application on computers.

Result

- To define an attribute NCE graph grammar.
- To formalize tabular forms based on an attribute NCE graph grammar.
- To show properties of the grammar for tabular forms.

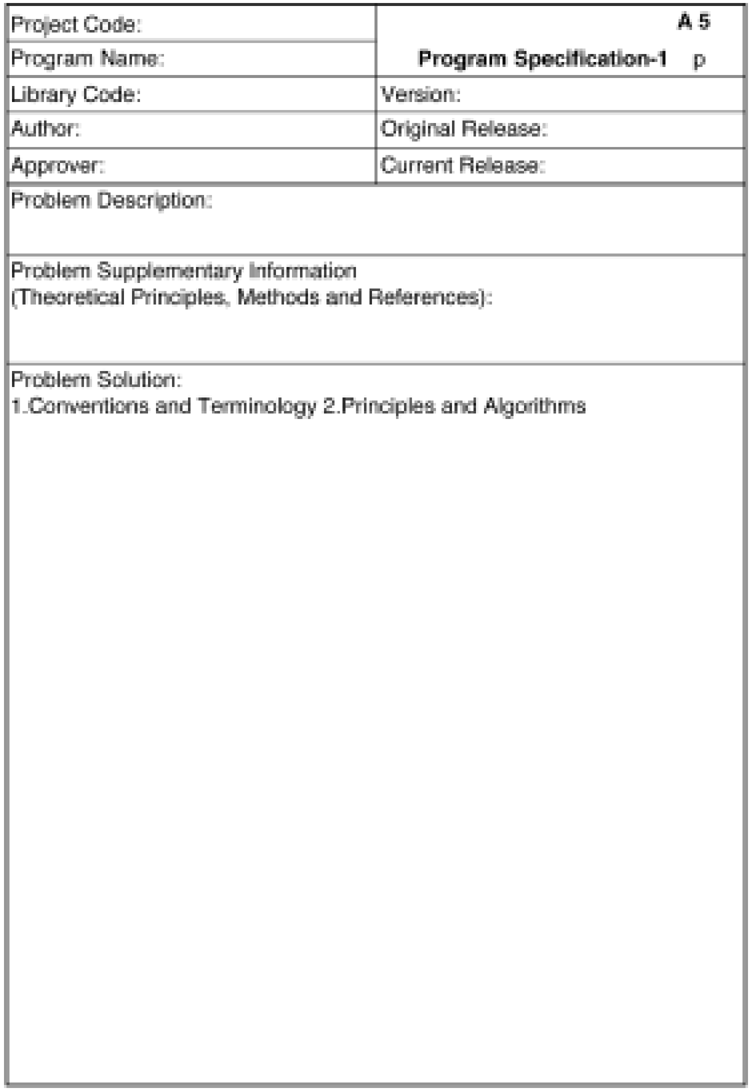
  context-free, precedence grammar

  280 productions, 1248 attribute rules
- To propose a parsing method for this grammar.

2. Program Documentation Language

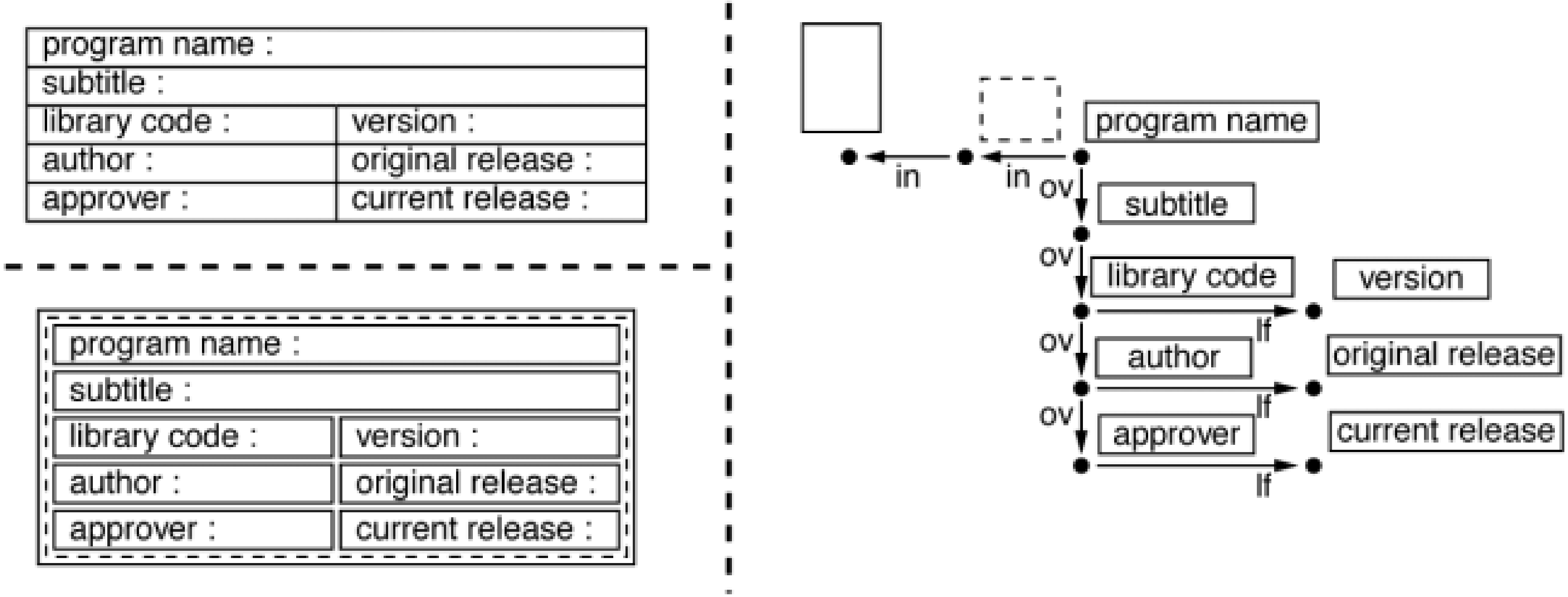
A program specification language Hiform

- 17types of Forms based on ISO6592
- A collection of tabular forms



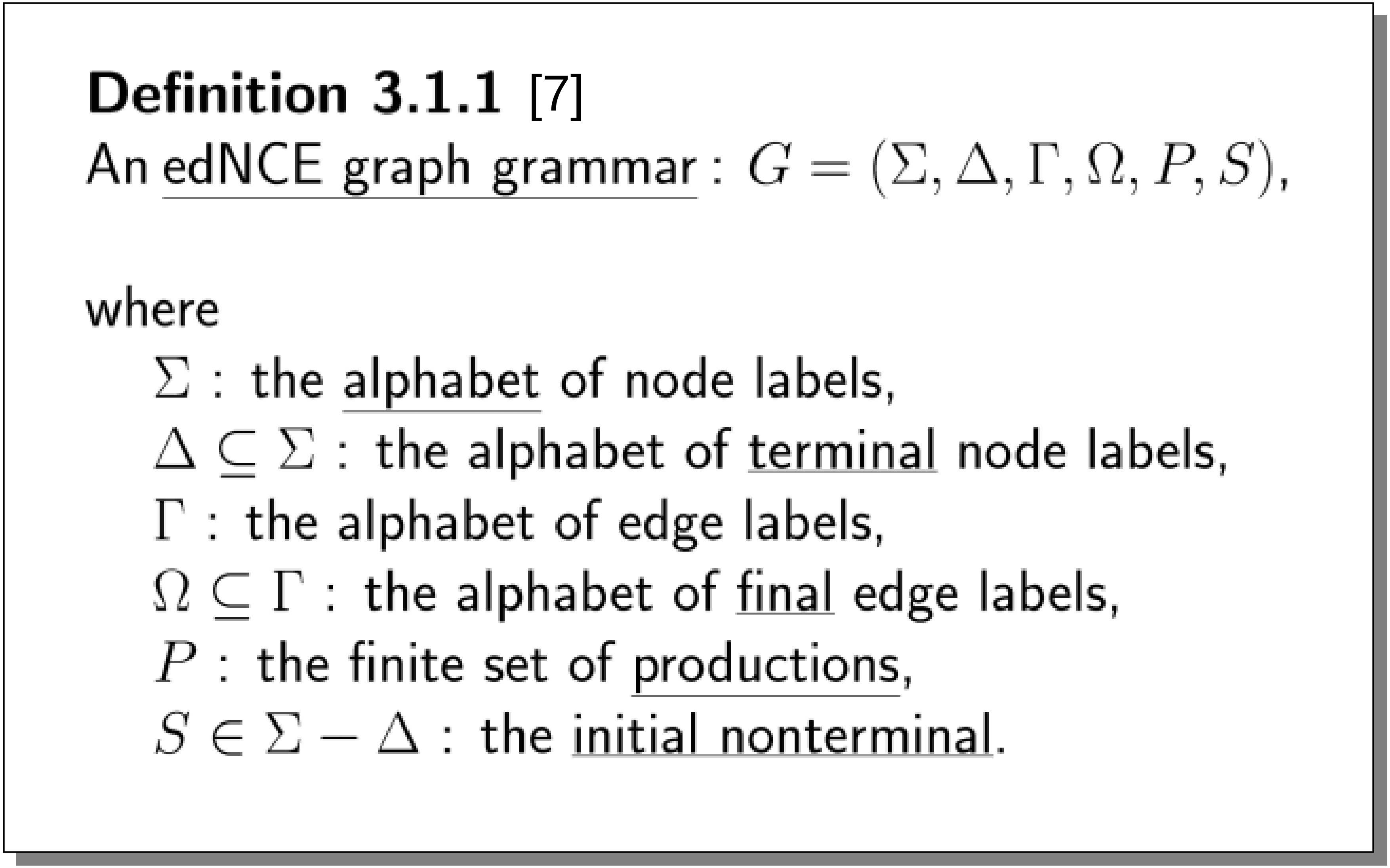
Project Code:	A 5 Program Specification-1 p
Program Name:	
Library Code:	Version:
Author:	Original Release:
Approver:	Current Release:
Problem Description:	
Problem Supplementary Information (Theoretical Principles, Methods and References):	
Problem Solution: 1.Conventions and Terminology 2.Principles and Algorithms	

Nested Diagram and Its Corresponding Marked Graph



3 Attribute NCE Graph Grammar

REVIEW



Definition 3.1.1 [7]

An edNCE graph grammar : $G = (\Sigma, \Delta, \Gamma, \Omega, P, S)$,

where

Σ : the alphabet of node labels,

$\Delta \subseteq \Sigma$: the alphabet of terminal node labels,

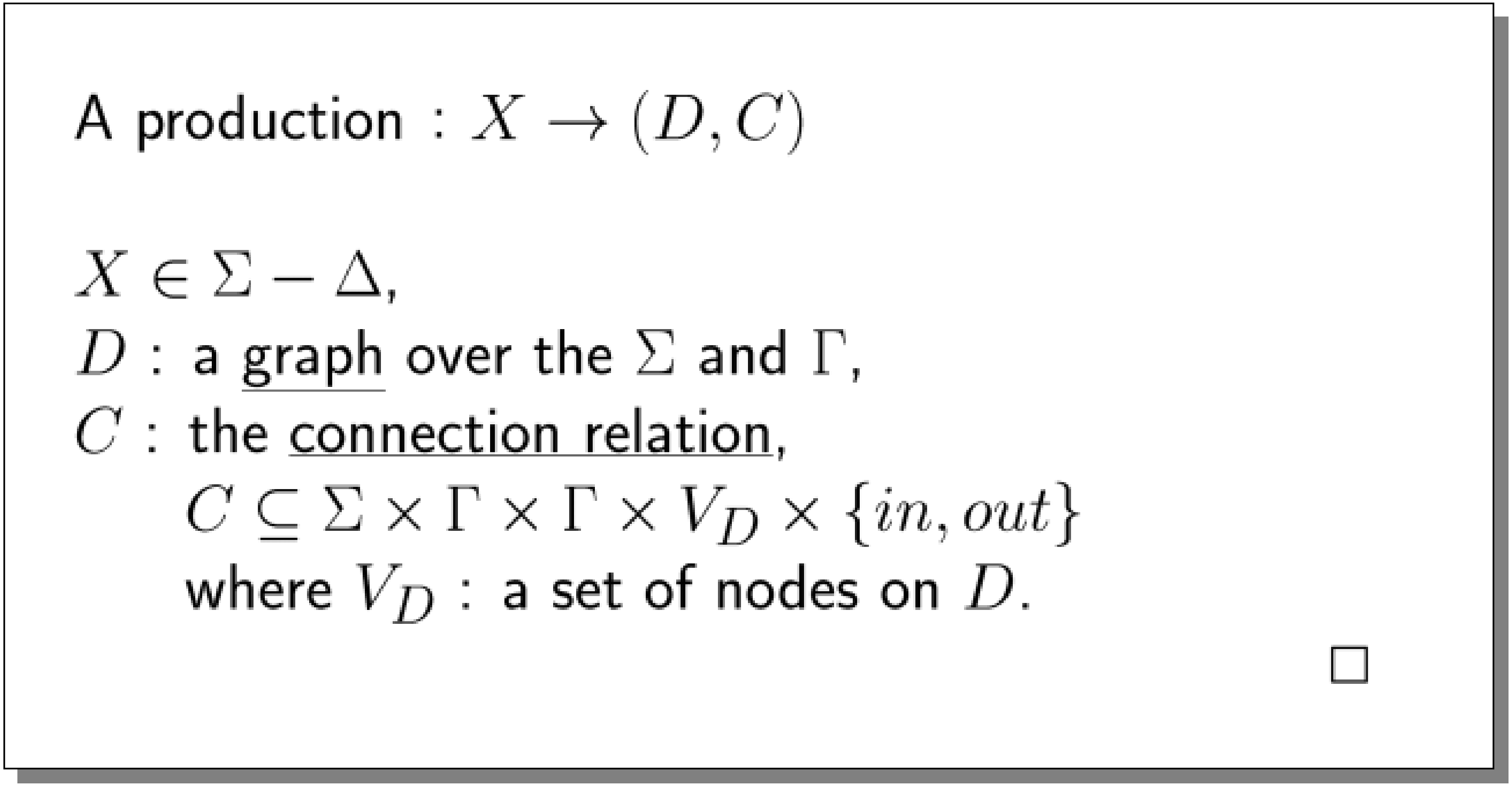
Γ : the alphabet of edge labels,

$\Omega \subseteq \Gamma$: the alphabet of final edge labels,

P : the finite set of productions,

$S \in \Sigma - \Delta$: the initial nonterminal.

REVIEW



A production : $X \to (D, C)$

$X \in \Sigma - \Delta$,

D : a graph over the Σ and Γ,

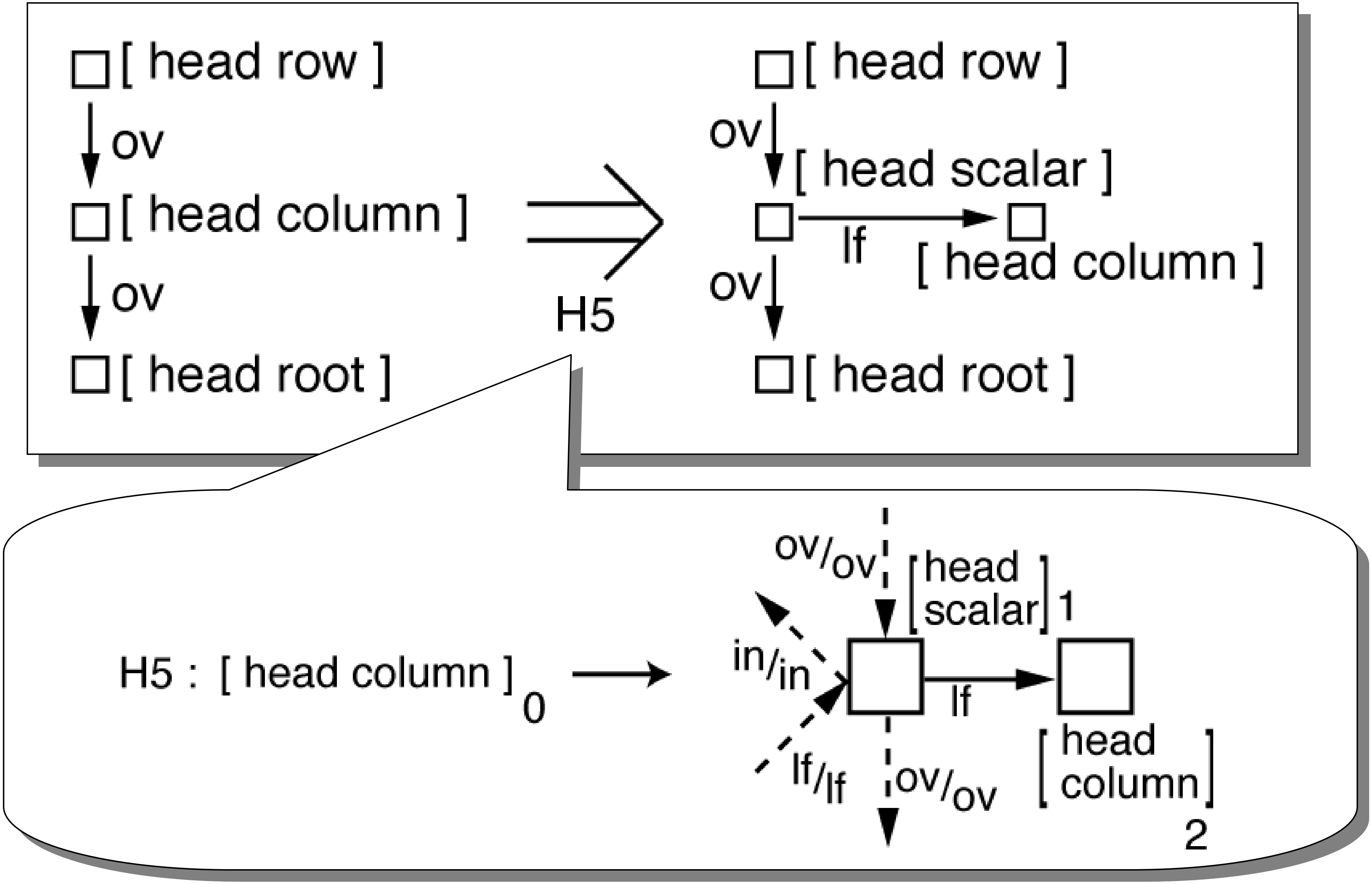
C : the connection relation,

$C \subseteq \Sigma \times \Gamma \times \Gamma \times V_D \times \{in, out\}$

where V_D : a set of nodes on D.

□

Rewrite a graph by production



□ [head row]

ov

□ [head column]

ov

□ [head root]

H5

□ [head row]

ov

[head scalar]

□ lf □

[head column]

ov

□ [head root]

H5 : [head column]$_0$ →

$^{ov}/_{ov}$

$\begin{bmatrix}\text{head}\\\text{scalar}\end{bmatrix}_1$

$^{in}/_{in}$

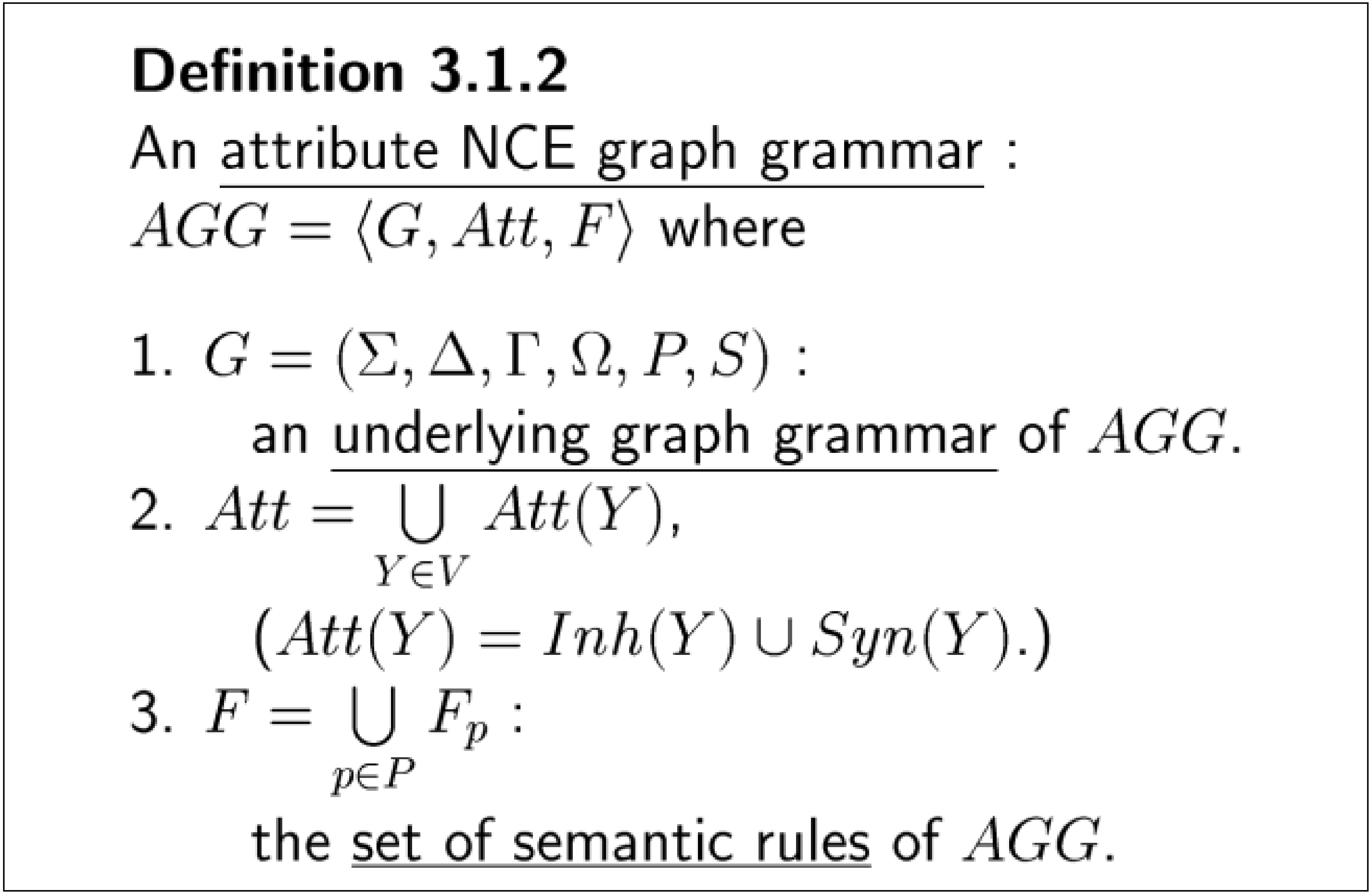
lf

$^{lf}/_{lf}$

$^{ov}/_{ov}$

$\begin{bmatrix}\text{head}\\\text{column}\end{bmatrix}_2$

Our Result 1

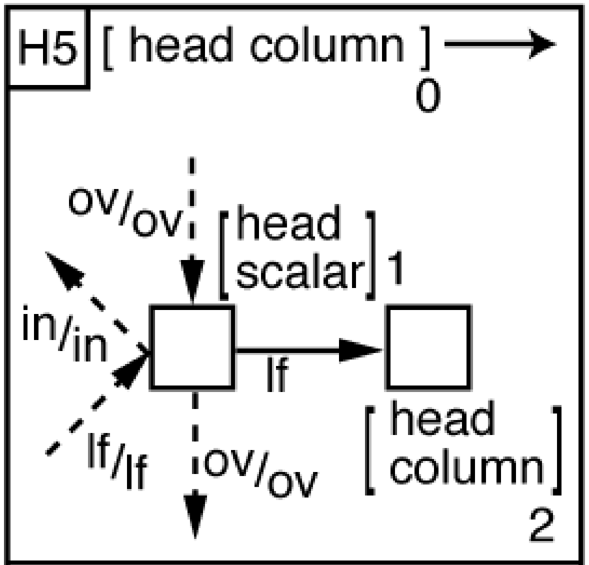


Definition 3.1.2

An attribute NCE graph grammar :
$AGG = \langle G, Att, F \rangle$ where

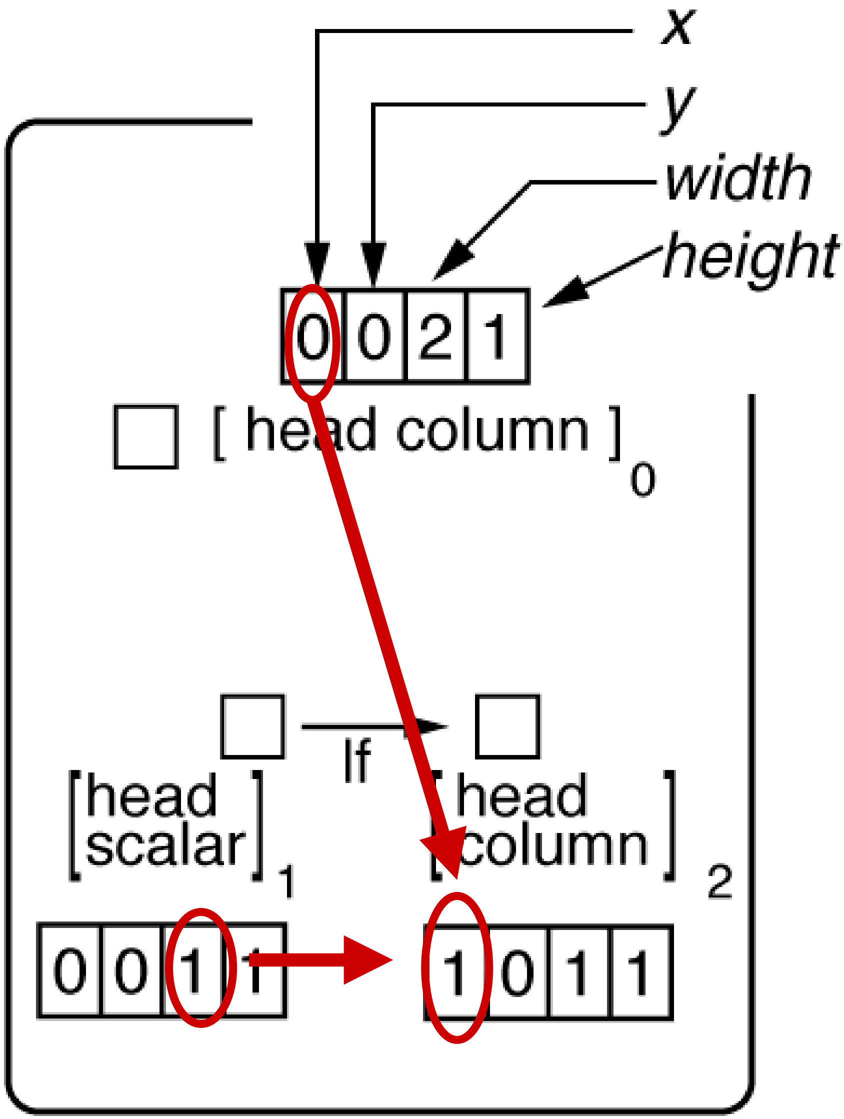
1. $G = (\Sigma, \Delta, \Gamma, \Omega, P, S)$:
   an underlying graph grammar of AGG.
2. $Att = \bigcup_{Y \in V} Att(Y)$,
   $(Att(Y) = Inh(Y) \cup Syn(Y).)$
3. $F = \bigcup_{p \in P} F_p$:
   the set of semantic rules of AGG.

Production with Attribute Rules



x(1) = x(0),
x(2) = x(0)+width(1),
y(1) = y(0), y(2) = y(0),
width(0)
= width(1)+width(2),
height(0) =
max(height(1)+height(2))

Production ‘H5’



sub-derivation tree

3.2 Precedence Relations

modify

Notation [cf. 1]

For every $m \in \Gamma$ and $\forall \# \in \Sigma$ let

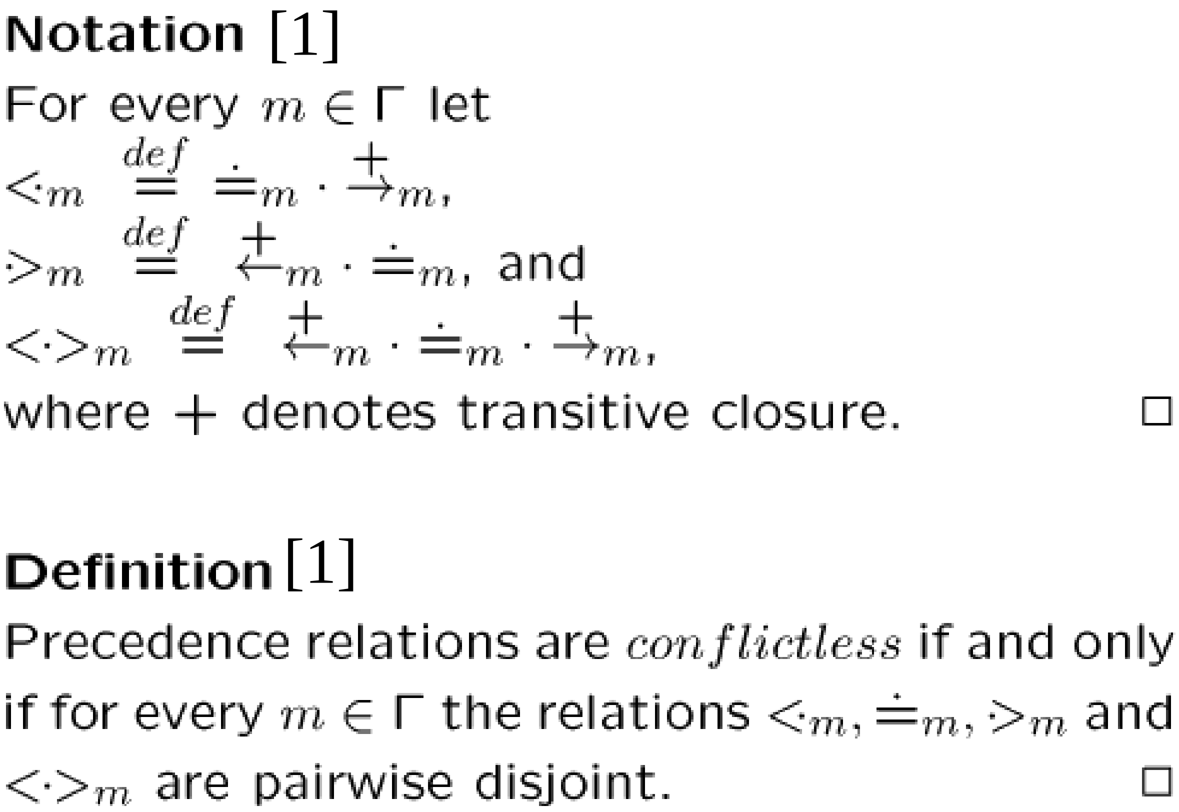
$$\doteq_m \overset{def}{=} \left\{ (A,B) \,\middle|\, \begin{array}{l} P \ni p : X \to (D,C), \\ \text{there is an edge } (x,y) \text{ on } D \\ \text{where } x \text{ is marked } A, \\ y \text{ is marked } B \text{ and} \\ (x,y) \text{ is labeled } m. \end{array} \right\}$$

$$\rightarrow_m \overset{def}{=} \left\{ (A,B) \,\middle|\, \begin{array}{l} P \ni p : A \to (D,C), \\ C \ni (\#, m, m, x, in), \\ \text{and the mark of } x \text{ is } B. \end{array} \right\}$$

$$\leftarrow_m \overset{def}{=} \left\{ (B,A) \,\middle|\, \begin{array}{l} P \ni p : A \to (D,C), \\ C \ni (\#, m, m, y, out), \\ \text{and the mark of } y \text{ is } B. \end{array} \right\}$$

□

REVIEW



Notation [1]

For every $m \in \Gamma$ let

$\lessdot_m \overset{def}{=} \doteq_m \cdot \xrightarrow{+}_m,$

$\gtrdot_m \overset{def}{=} \xleftarrow{+}_m \cdot \doteq_m,$ and

$<\!\!\cdot\!\!>_m \overset{def}{=} \xleftarrow{+}_m \cdot \doteq_m \cdot \xrightarrow{+}_m,$

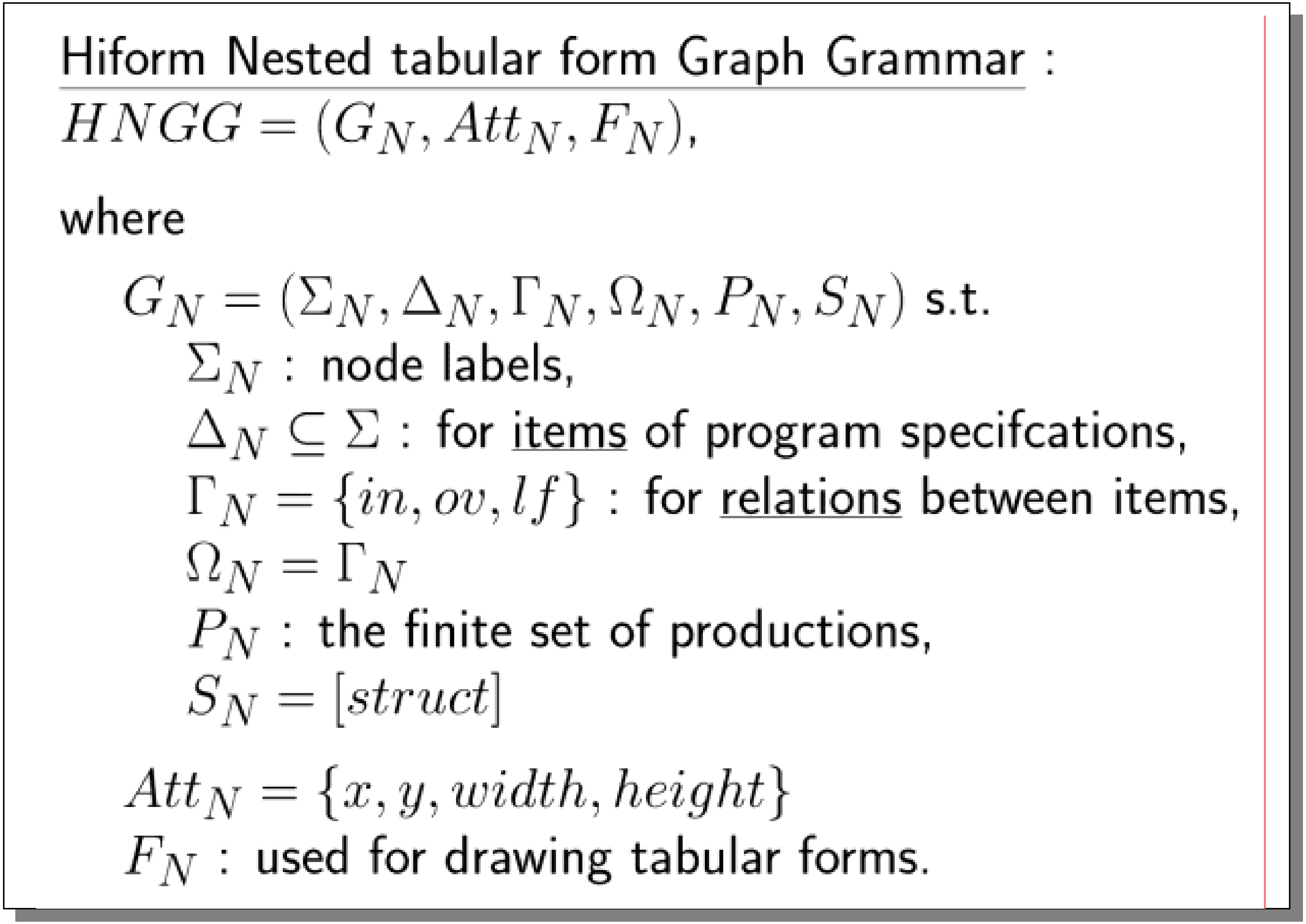
where $+$ denotes transitive closure. □

Definition [1]

Precedence relations are $conflictless$ if and only if for every $m \in \Gamma$ the relations $\lessdot_m, \doteq_m, \gtrdot_m$ and $<\!\!\cdot\!\!>_m$ are pairwise disjoint. □

4 AN ATTRIBUTE GRAPH GRAMMAR FOR HIFORM

Our Result 2



Hiform Nested tabular form Graph Grammar :

$HNGG = (G_N, Att_N, F_N)$,

where

$G_N = (\Sigma_N, \Delta_N, \Gamma_N, \Omega_N, P_N, S_N)$ s.t.

Σ_N : node labels,

$\Delta_N \subseteq \Sigma$: for items of program specifcations,

$\Gamma_N = \{in, ov, lf\}$: for relations between items,

$\Omega_N = \Gamma_N$

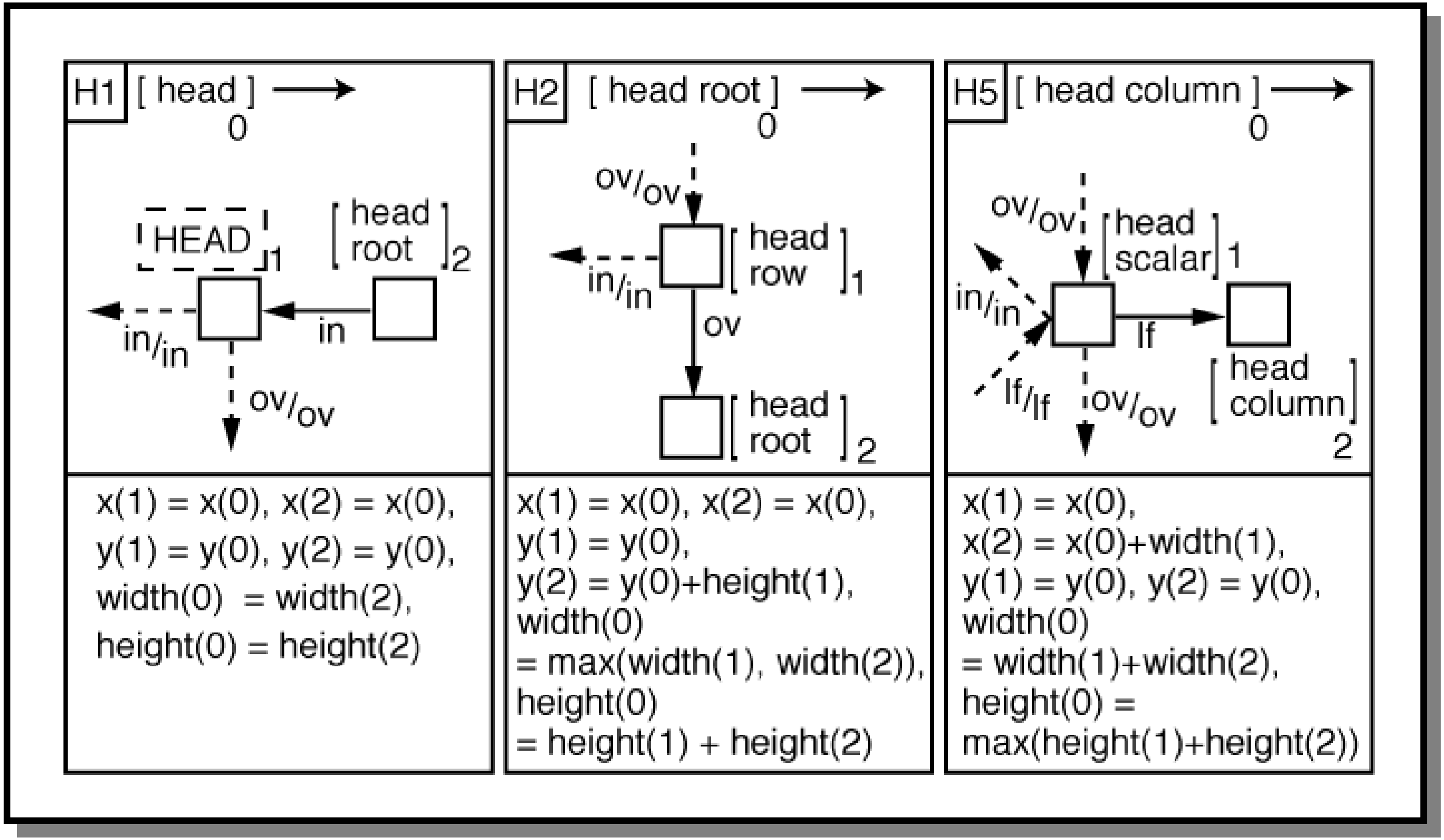
P_N : the finite set of productions,

$S_N = [struct]$

$Att_N = \{x, y, width, height\}$

F_N : used for drawing tabular forms.

Productions of HNGG



x(1) = x(0), x(2) = x(0),
y(1) = y(0), y(2) = y(0),
width(0) = width(2),
height(0) = height(2)

x(1) = x(0), x(2) = x(0),
y(1) = y(0),
y(2) = y(0)+height(1),
width(0)
= max(width(1), width(2)),
height(0)
= height(1) + height(2)

x(1) = x(0),
x(2) = x(0)+width(1),
y(1) = y(0), y(2) = y(0),
width(0)
= width(1)+width(2),
height(0) =
max(height(1)+height(2))

- **Features of HNGG**

GG	Type	Rewriting Rule	Attribute Rule
HNGG	Context-free	280	1248

Our Result 3

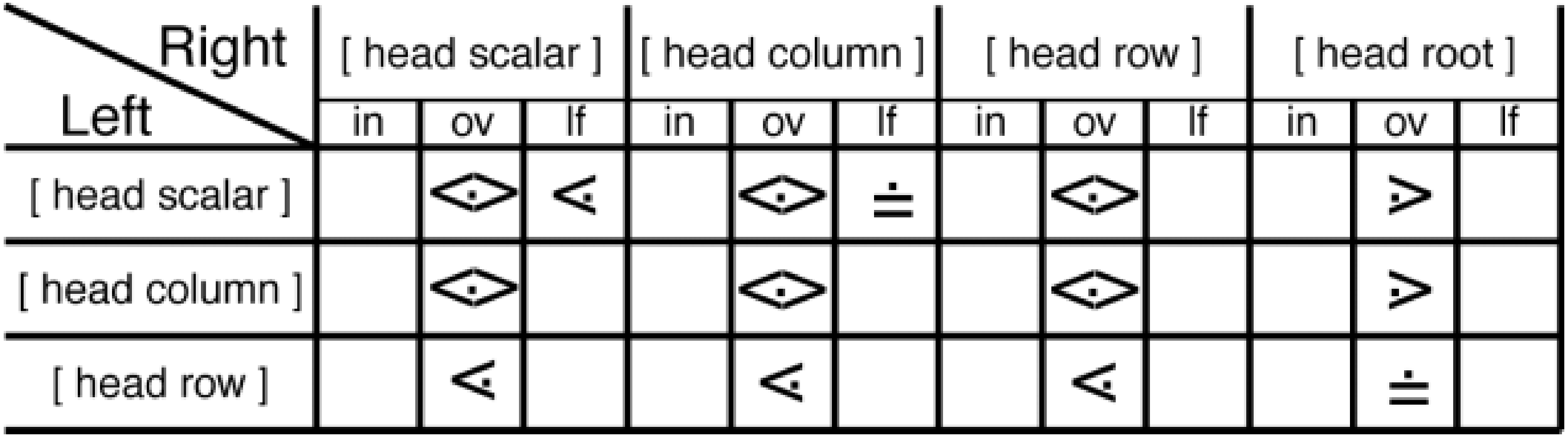
Proposition

HNGG is a precedence graph grammar.

Proof.

We construct 5376 precedence relations. The relations are shown to be pairwise disjoint. □

Fig. A part of precedence relations of HNGG



Right / Left	[head scalar]			[head column]			[head row]			[head root]		
	in	ov	lf	in	ov	lf	in	ov	lf	in	ov	lf
[head scalar]		<·>	<·		<·>	≐		<·>			·>	
[head column]		<·>			<·>			<·>			·>	
[head row]		<·			<·			<·			≐	

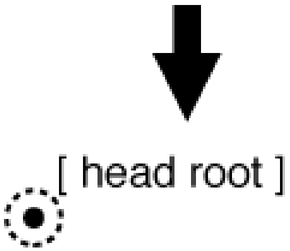
5 Paring of Precedence Attribute Graph Grammars

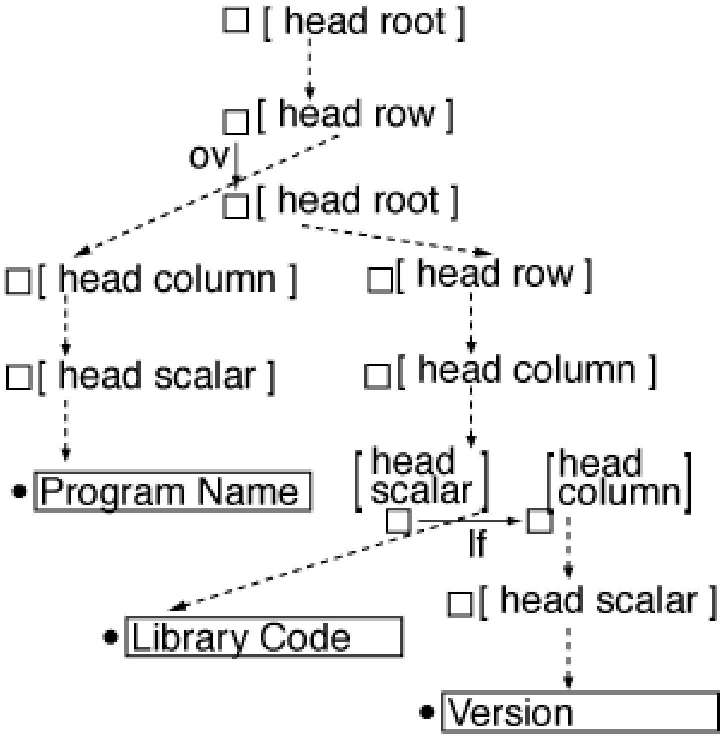
Our Result 4.

5.1 Syntactic Analysis

- The syntactic analysis is done by parsing of precedence graph language.
  - use a precedence property.

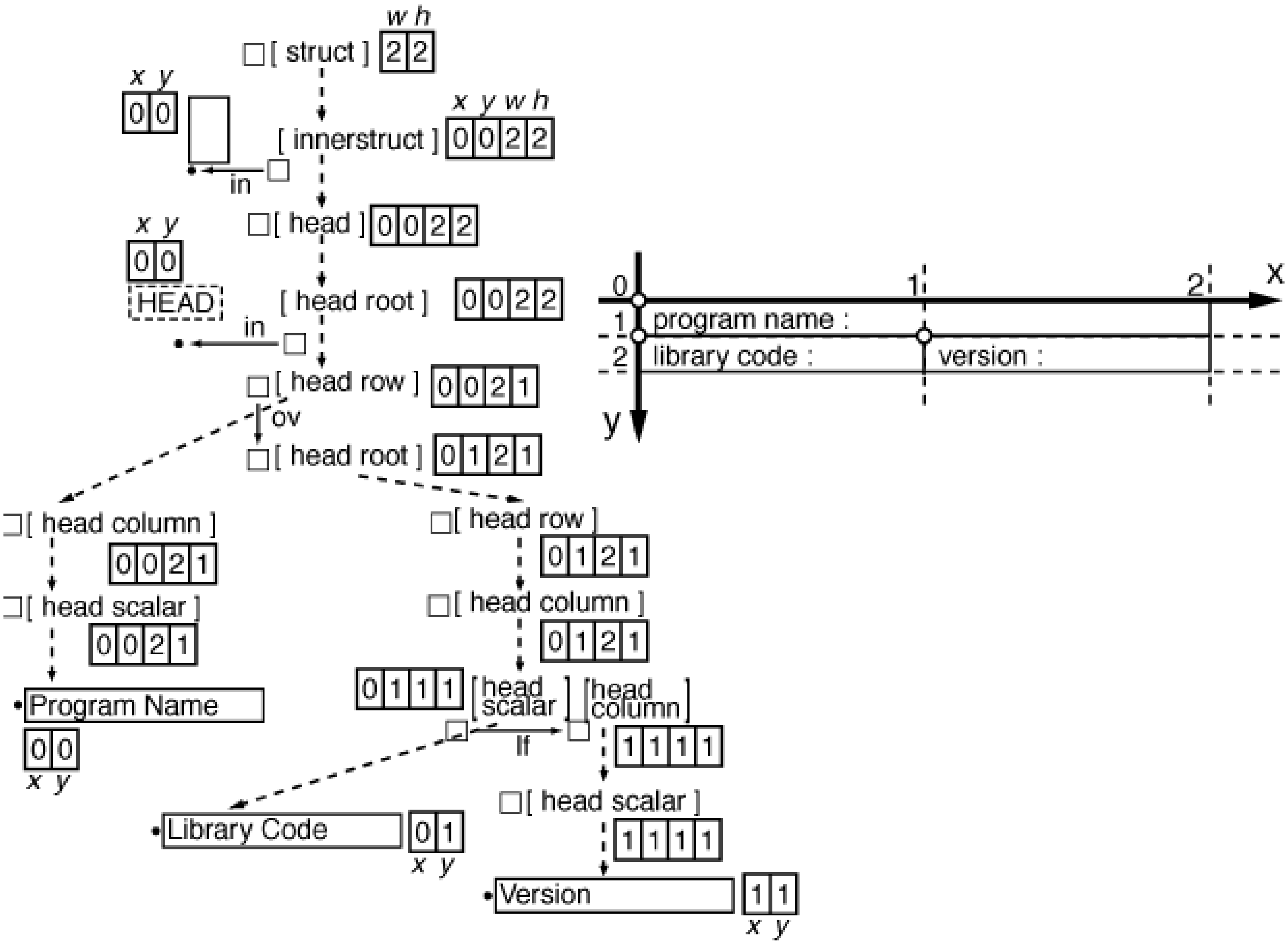
How to use precedence rule





- Input Graph
- Derivation Tree

5.2 Attribute Evaluation



6 Conclusion

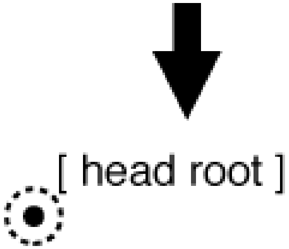
- We proposed an attribute NCE graph grammar for tabular forms based on ISO6592.
  - ●rewriting rule : logical structure
  - ●attribute rule : visual structure

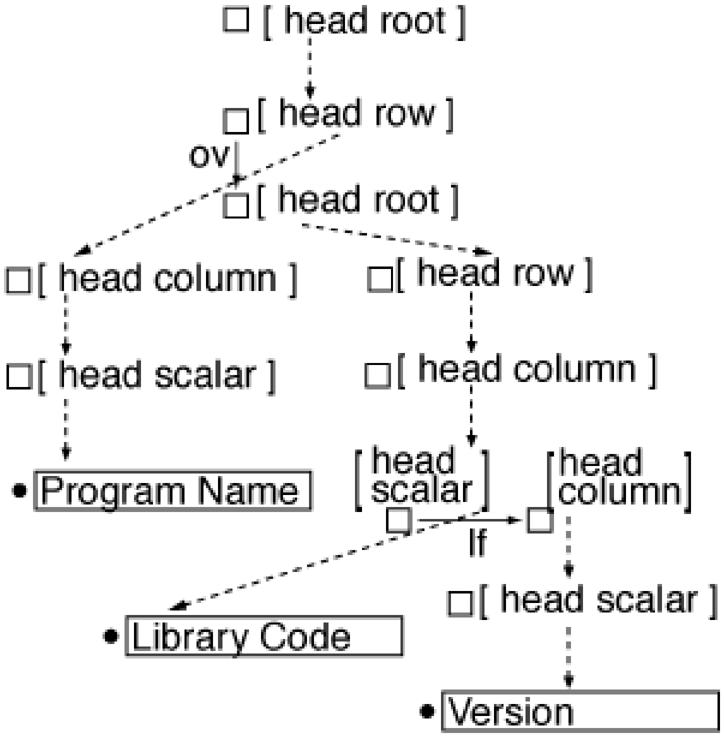
Graph Grammar	rewriting rules	attribute rules
HNGG	**280**	**1248**

- We propose a parsing method by using precedence relations.

Our Result 4.

How to use precedence rule




□ [head root]
□[head row]
ov
□[head root]
□[head column]
□[head row]
□[head scalar]
□[head column]
● Program Name
[head scalar]
[head column]
□
□
lf
□[head scalar]
● Library Code
● Version