Application of Attribute Graph Grammars to Syntactic Editing of Tabular Forms

Kiyonobu TOMIYAMA (Nihon Univ.)
○Tomokazu ARITA (Nihon Univ.)
Kensei TSUCHIDA(Toyo Univ.)
Takeo YAKU(Nihon Univ.)

Terms

- **Tabular forms** with **syntax**
- **Graph grammars** for tabular form syntax
- **Attribute** : used for drawing
- **edNCE** : edge-direct Neighbourhood Controlled Embedding

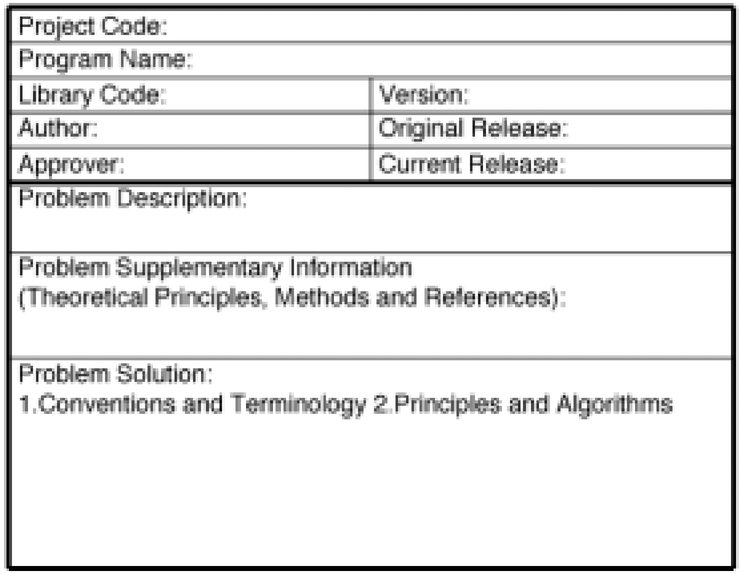
  (cf. NLC : Node Labeled Controlled) :

  A type of embedding mechanisms of right hand side graph into host graph for **graph rewriting**
- **Syntactic editing** : editing defined by sequence of rewriting rules

Types of Visual Data (ex. Power Point)

- Character String
- Tree Diagrams
- **Tabular Forms (Our Target)**
  - **Modular Tabular Form**
  - **Tessellation Tabular Form**
- Graphs
- Images

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

Modular Tabular Form

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

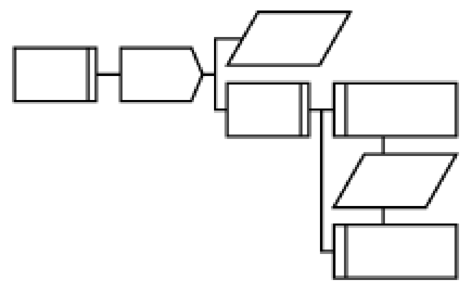
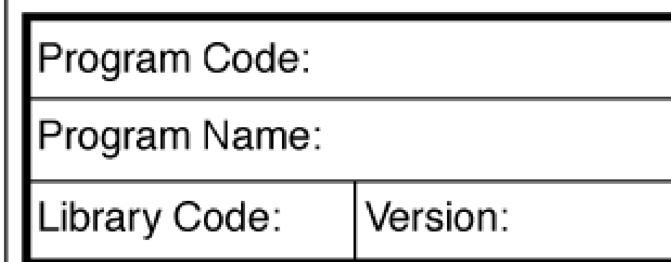
Tessellation Tabular Form

Goals

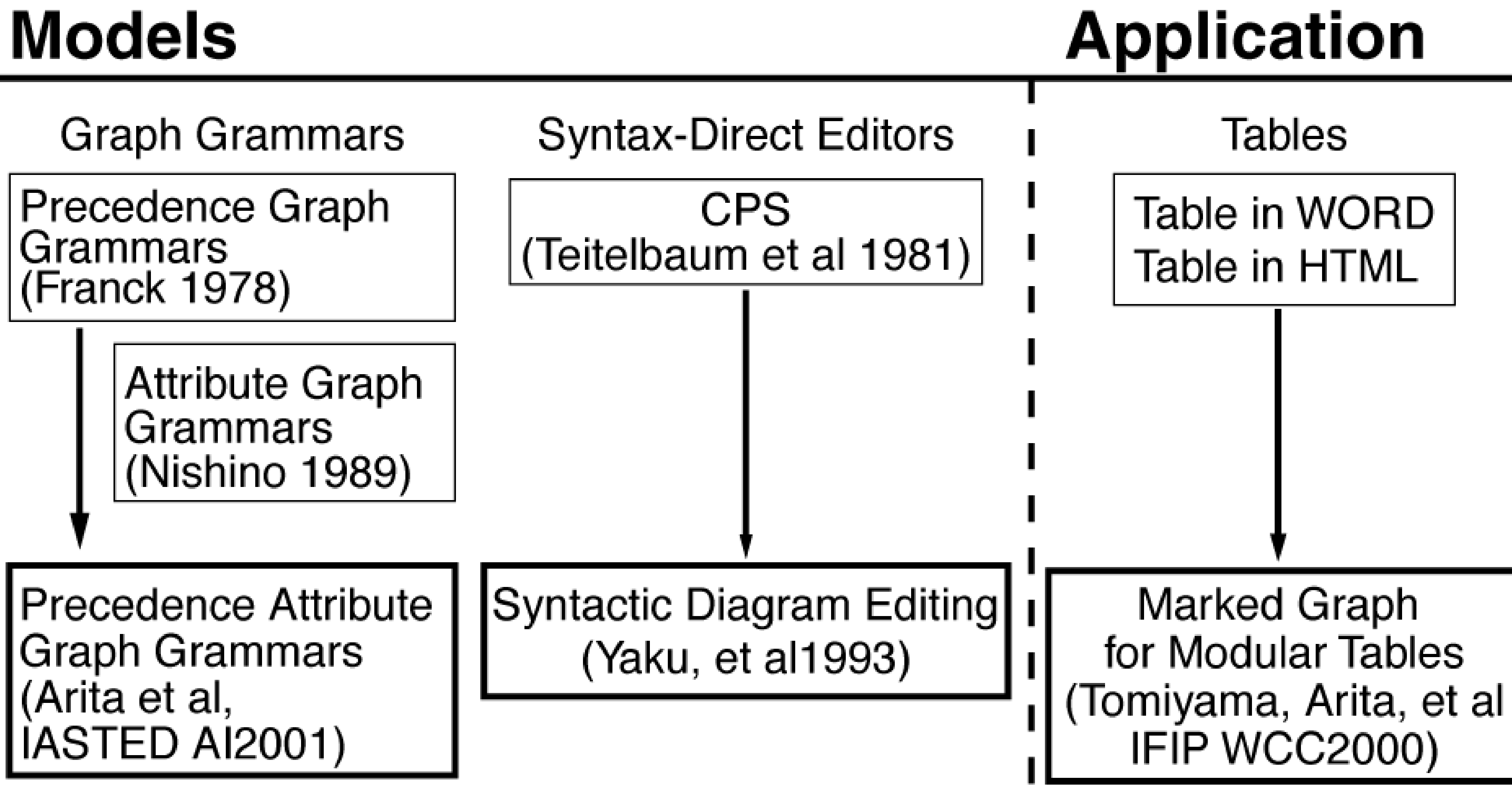
Constructing the **syntactic editing mechanisms** of tabular forms based on attribute graph grammar.

1. Introduction

Positions of this Paper

	CPS	Flowcharts	Program Specification
Diagrams	string	Hierarchical Diagram 	Modular Diagram 
Represent-ation	string	Attribute Tree	Attribute Marked Tree
Syntax	CFG	Attribute CFGG (Yaku, Tsuchida et al)	Attribute NCE CFGG (Arita et al)
Editor Command	○	○ (Yaku, Tsuchida, et al)	**This Paper**

Background


Models
Application
Graph Grammars
Syntax-Direct Editors
Tables
Precedence Graph Grammars (Franck 1978)
CPS (Teitelbaum et al 1981)
Table in WORD
Table in HTML
Attribute Graph Grammars (Nishino 1989)
Precedence Attribute Graph Grammars (Arita et al, IASTED AI2001)
Syntactic Diagram Editing (Yaku, et al1993)
Marked Graph for Modular Tables (Tomiyama, Arita, et al IFIP WCC2000)

Why Graph Grammars

- Tabular Forms have syntax
- To determine the scope of rewriting by cell deletion or insertion
- Without graph grammars, often to collapse whole structure of tables

Why Attribute Graph Grammars

- Rewriting scope determined by graph grammars.
- Attribute rules concerned to graph grammars
- Incremental drawing

Motivation

- To investigate whether graph grammars can effectively formalize table processing.

Purpose

- Formalization of editing by graph grammars
- Properties of editing model
- Algorithms of editing by graph grammars
- Attribute evaluation for drawing

Results

- Definitions of deletion, insertion by rewriting rules of edNCE graph grammars
- Confluence of editing
- Linear time algorithm with attribute evaluation for primitive drawing conditions

Related Works

- Related works for **syntactic editing methods are CPS, DIAGEN(Minas et al.)** and so on.

2. Preliminaries

2. Preliminaries

program specification language

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

program name : Hanoi_main	**A**
subtitle : hanoi	**General document**
library code : cs - 2000 - 01	**version :** 1.0
author : Kiyonobu Tomiyama	**original release :** 1999/12/22
approve :	**current release :** 2000/01/28
key words : Hanoi Tower	**CR-code :**
scope : Fundamental	
variant :	
language : Java	**software req. :** JDK 1.2
operation : Interactive batch realtime	**hardware req. :**
references :	

function : 1. list and explanation of input data or parameter,
2. list and explanation of output data or return value.

1. list and explanation of input data.

```
int n;              [ Number of Plates ]
String target;      [ Target Symbol ]
String work;        [ Working Symbol ]
String destination; [ Destination Symbol ]
```

2. list and explanation of output data and return value.

```
output data  : No. to be moved: Source Symbol -> Destination Symbol
return value : void
```

example :

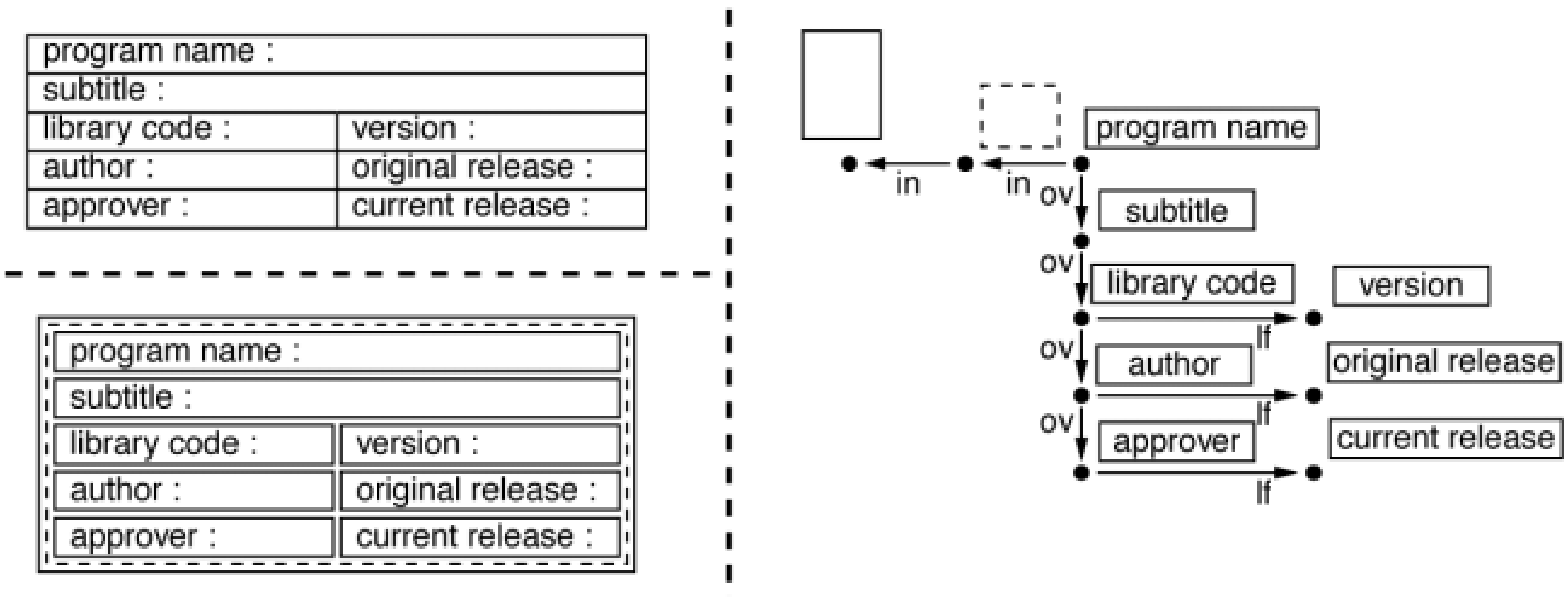
1. Example of Operation

```
hanoi(5, A, B, C)
```

2. Example of Output

```
1: A -> C
2: A -> B
1: C -> B
3: A -> C
1: B -> A
. . . . . .
```

Modular Tabular Form and Its Corresponding Marked Graph



2.1 An Attribute edNCE Graph Grammar

2.1.1 Definition

An attribute NCE graph grammar is a 3–tuple $AGG = \langle G, Att, F \rangle$

1. $G = (\Sigma, \Delta, \Gamma, \Omega, P, S)$ is an underlying graph grammar of AGG.

2. $Att = \bigcup_{Y \in V} Att(Y)$,
   $(Att(Y) = Inh(Y) \cup Syn(Y).)$

3. $F = \bigcup_{p \in P} F_p$ is the set of semantic rules of AGG.

2.1 edNCE Graph Grammar[6]

Definition 2.1.1

edNCE graph grammar：G=（$\boldsymbol{\Sigma}$, $\boldsymbol{\Delta}$, $\boldsymbol{\Gamma}$, $\boldsymbol{\Omega}$,P,S） where

$\boldsymbol{\Sigma}$：The alphabet of node labels

$\boldsymbol{\Delta} \subseteq \boldsymbol{\Sigma}$：The alphabet of terminal node labels

$\boldsymbol{\Gamma}$：The alphabet of edge labels

$\boldsymbol{\Omega} \subseteq \boldsymbol{\Delta}$： The alphabet of final edge labels

P： The finite set of productions

S$\in \boldsymbol{\Sigma} - \boldsymbol{\Delta}$： The initial nonterminal

edNCE Graph Grammar (continued)

production $p : X \rightarrow (D,C)$

$X \in \Sigma - \Delta$

$(D,C) \in GRE_{\Sigma,\Gamma}$

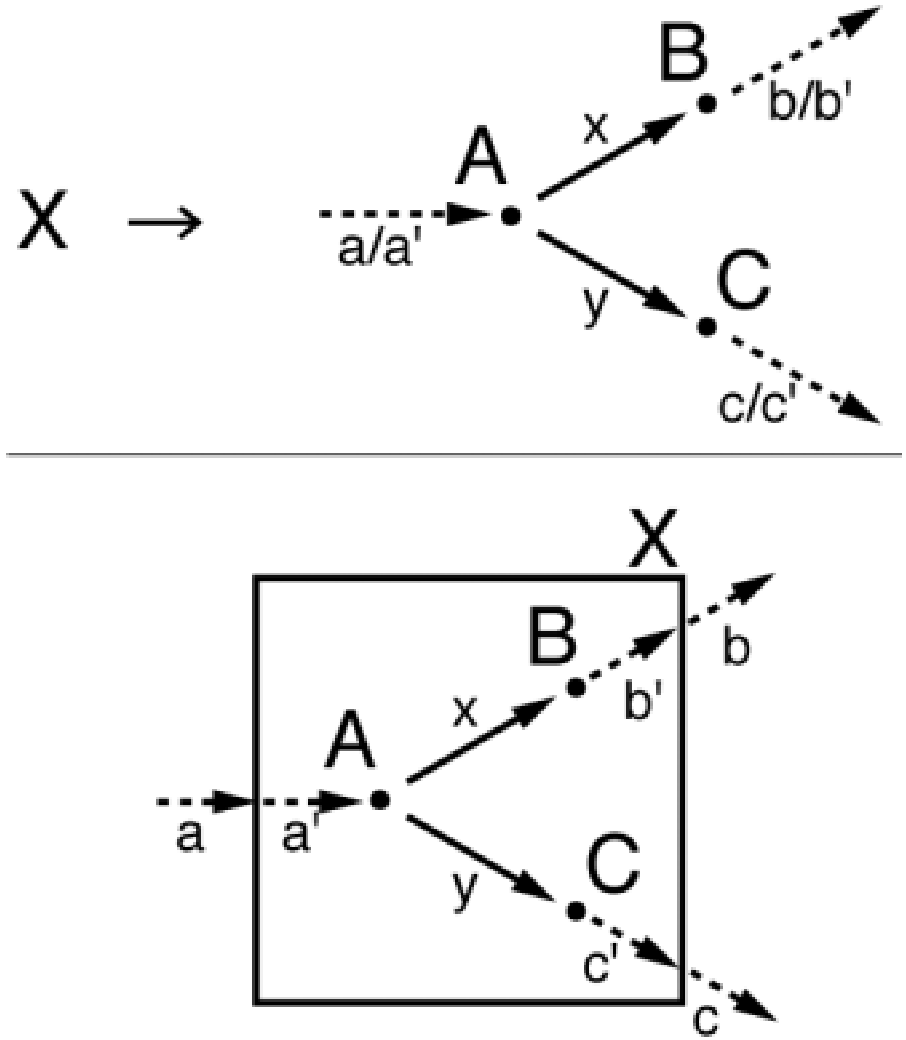
$D \in GR_{\Sigma,\Gamma}$

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

: connection relation

□

Production



2.2 COMPOSITION OF PRODUCTION COPIES [4]

2.2.1 Definition

Let $G=(\boldsymbol{\Sigma}, \boldsymbol{\Delta}, \boldsymbol{\Gamma}, \boldsymbol{\Omega}, P, S)$：edNCE-CFGG ,

$p_1:X_1 \rightarrow (D_1,C_1)$, $p_2:X_2 \rightarrow (D_2,C_2)$：production copy of G,

for X_2 exists in node labels of D_1

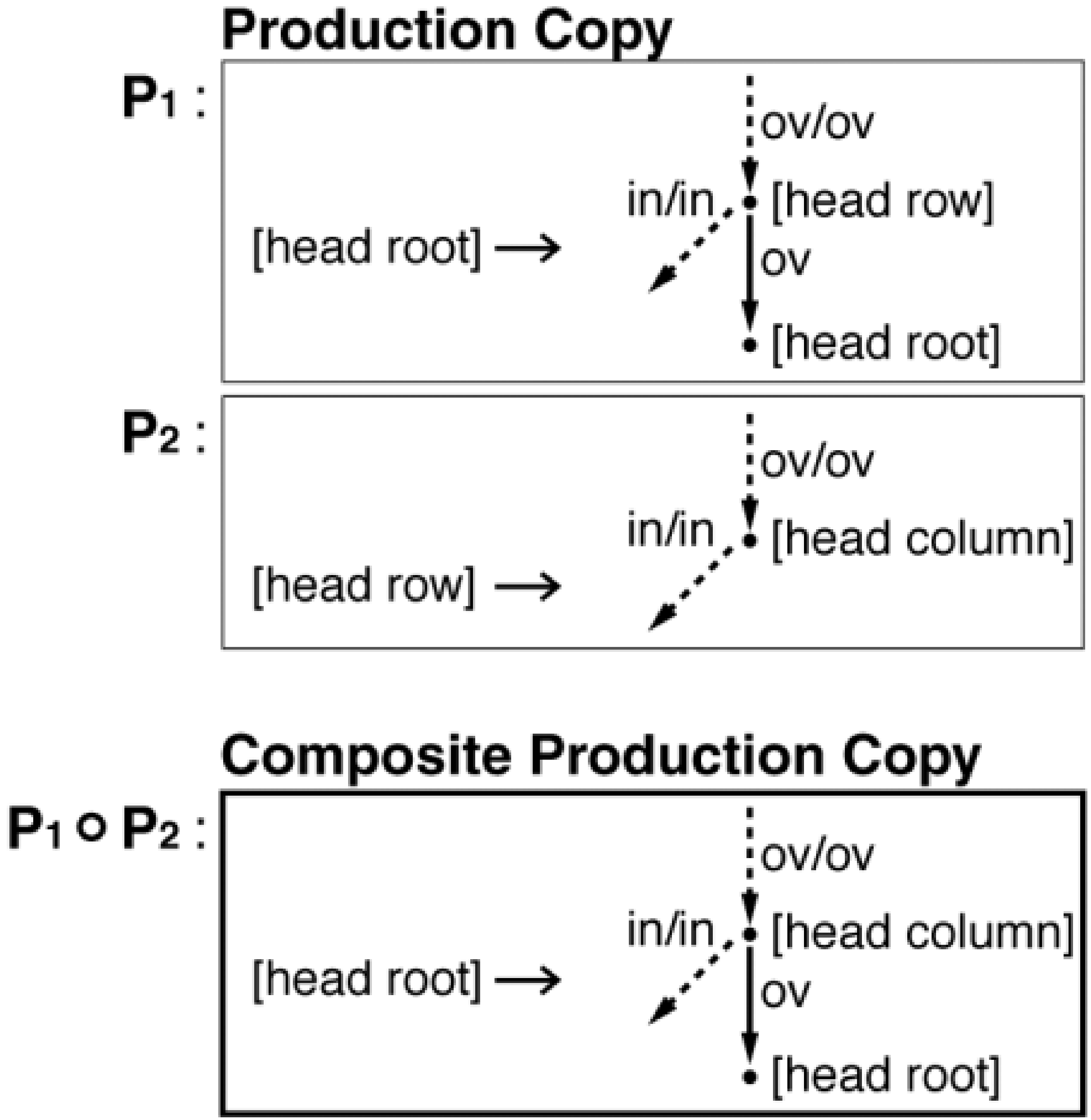
The <u>composite production copy</u> $p: X_1 \rightarrow (D,C)$, where

$D=D_1-\{X_2\} \cup D_2$

$C=\{(\boldsymbol{\sigma}, \boldsymbol{\beta}/\boldsymbol{\gamma}, \boldsymbol{\omega}, d) \in C1 \mid \boldsymbol{\omega} \in V_{D1}-V_{X2}\}$

$\cup \{(\boldsymbol{\sigma}, \boldsymbol{\beta}/\boldsymbol{\delta}, y, d) \mid \exists \boldsymbol{\omega} \in V_{X2} \exists \boldsymbol{\gamma} \in \boldsymbol{\Gamma}, (\boldsymbol{\sigma}, \boldsymbol{\beta}/\boldsymbol{\gamma}, \boldsymbol{\omega}, d) \in C_1, (\boldsymbol{\sigma}, \boldsymbol{\gamma}/\boldsymbol{\delta}, y, d) \in C_2\}$ □

Example (Composite Production copy)



2.3 Confluence Property [6]

Definition 2.3.1

An edNCE graph grammar G=（Σ，Δ，Γ，Ω,P,S） is <u>confluent</u> if the following holds for every sentential form H of G:

If H $\Rightarrow_{u1,p1}$ H1 $\Rightarrow_{u2,p2}$ H12 and H $\Rightarrow_{u2,p2}$ H1 $\Rightarrow_{u1,p1}$ H21 (p1,p2∈P)

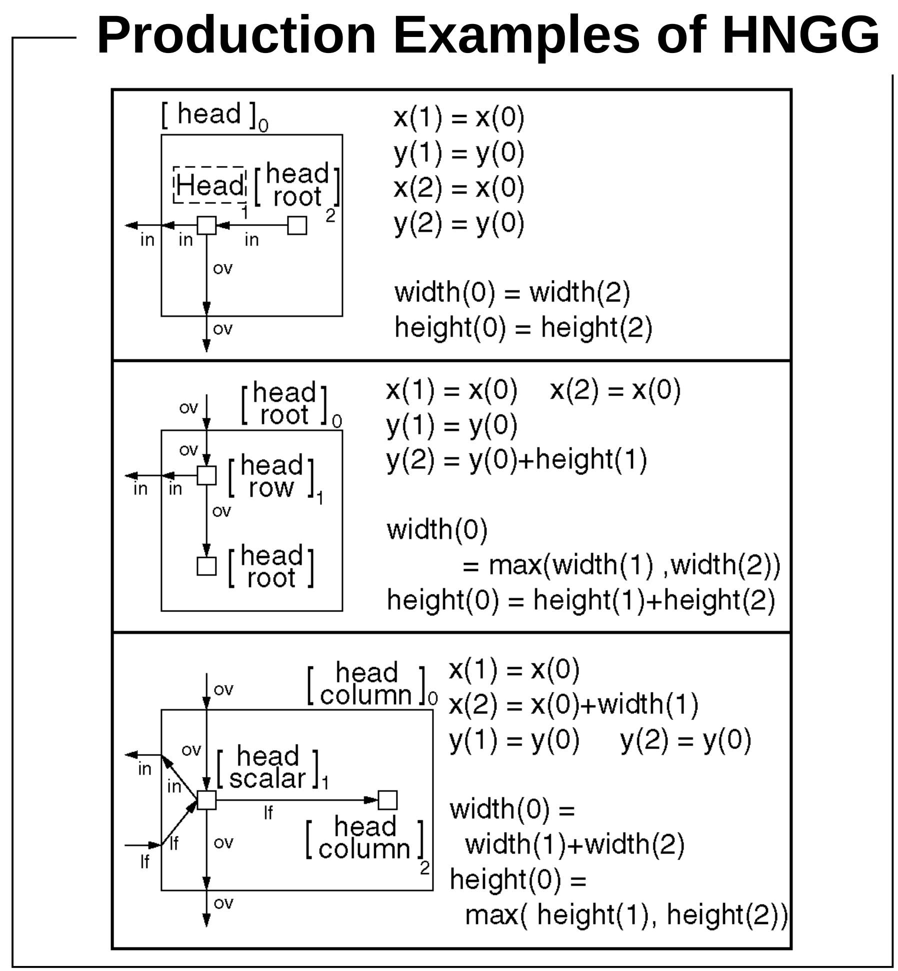
are derivation of G with

u1,u2∈VH and u1≠u2,

then H12=H21 □

HNGG (Hiform Nested Form Graph Grammar)

Graph grammar	HNGG
productions	280
attribute rules	1248
precedence relations	5376



3 Editing of Modular Tabular Form

3.1 Production Instance

3.1.1 Definition

A production instance is a 3-tuple (ω, p_i, H'_{p_i}), $\overleftrightarrow{def}$

1. $\omega \in V_{D_{i-1}}$ is a node removed during the derivation $D_{i-1} \Rightarrow_{p_i} D_i$.

2. $p_i : X_{p_i} \rightarrow (H_{p_i}, C_{p_i}) \in P$ is a production.

3. H'_{p_i} is an embedded graph isomorphic to H_{p_i} during $D_{i-1} \Rightarrow_{p_i} D_i$.

We denote $D_{i-1} \overset{\omega H'_{p_i}}{\underset{p_i}{\Longrightarrow}} D_i$
if D_{i-1} is directly derived D_i by applying the instance (ω, p_i, H'_{p_i}).

3.2 Syntactic Insertion

3.2 Syntactic Insertion

3.2.1 Definition

For an derivation sequence

$D_0 \overset{\omega_1 H'_{p_1}}{\Rightarrow}_{p_1} \cdots \overset{\omega_{i-1} H'_{p_{i-1}}}{\Rightarrow}_{p_{i-1}} D_{i-1} \overset{\omega H'_{p_i}}{\Rightarrow}_{p_i} D_i \overset{\omega_{i+1} H'_{p_{i+1}}}{\Rightarrow}_{p+1} \cdots \overset{\omega_n H'_{pn}}{\Rightarrow}_{p_n} D_n$

$(p_j : X_{p_j} \to (H_{p_j}, C_{p_j}), 1 \leq j \leq n)$, q is <u>insertable</u> (for p_i)

$\overset{\Longleftrightarrow}{def}$

there is an instance (ω, q, H'_q) $(q : X_q \to (H_q, C_q) \in P_N)$

s.t. $D_{i-1} \overset{\omega H'_q}{\Rightarrow}_q Q$,

there is a derivation sequence

s.t. $D_{i-1} \overset{\omega H'_q}{\Rightarrow}_q Q \overset{\omega' H'_{p_i}}{\Rightarrow}_{p_i} D'_i \overset{\omega_{i+1} H'_{p_{i+1}}}{\Rightarrow}_{p_{i+1}} \cdots \overset{\omega_n H'_{pn}}{\Rightarrow}_{p_n} D'_n$.

1. Trace the derivation sequence D_n back to D_{i-1}.

2. Apply the instance (ω, q, H'_q) to D_{i-1} , and obtain the resultant graph Q .

3. Apply the instance sequence ($(\omega'_i, p_i, H'_{p_i})$, $(\omega'_{i+1}, p_{i+1}, H'_{p_{i+1}})$, $\cdots$, $(\omega'_n,\ p_n,\ H'_{p_n})$) to Q, and get the resultant graph D'_n.

□

insertable by composite production copy is similarly defined.

3.2.3 Definition

A graph H' is obtained by syntactic insertion of a graph A at an edge x in a graph H.

$\overset{\Longleftrightarrow}{def}$

1. A composite production copy q for the graph A and an edge x exists.
2. There exists an instance sequence i_q for q and an instance sequence i_H for H. An instance sequence S is obtained by insertion of i_q into i_H.
3. H' is derived by S.

□

3.2.4 Proposition
Let H be the graph obtained from G by the insertion of nodes a and b at edge x and edge y respectively in this order, in HNGG. Let H' be the graph obtained from G by the insertion of nodes b and a at edge y and edge x respectively in this order, in HNGG.
$\Longleftrightarrow$

$H = H'$.

Proof HNGG has confluence property. Thus, the theorem is verified.

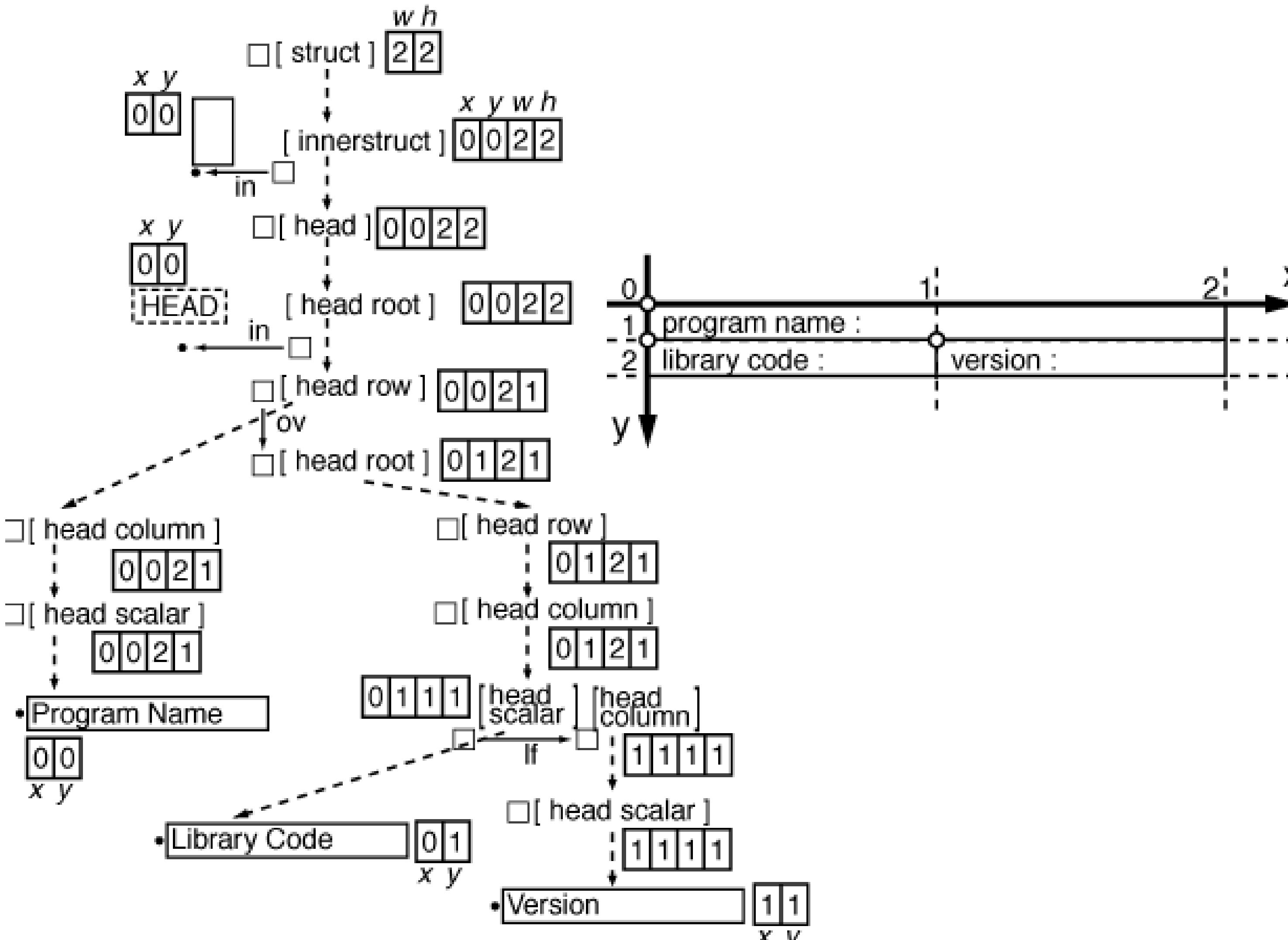
□

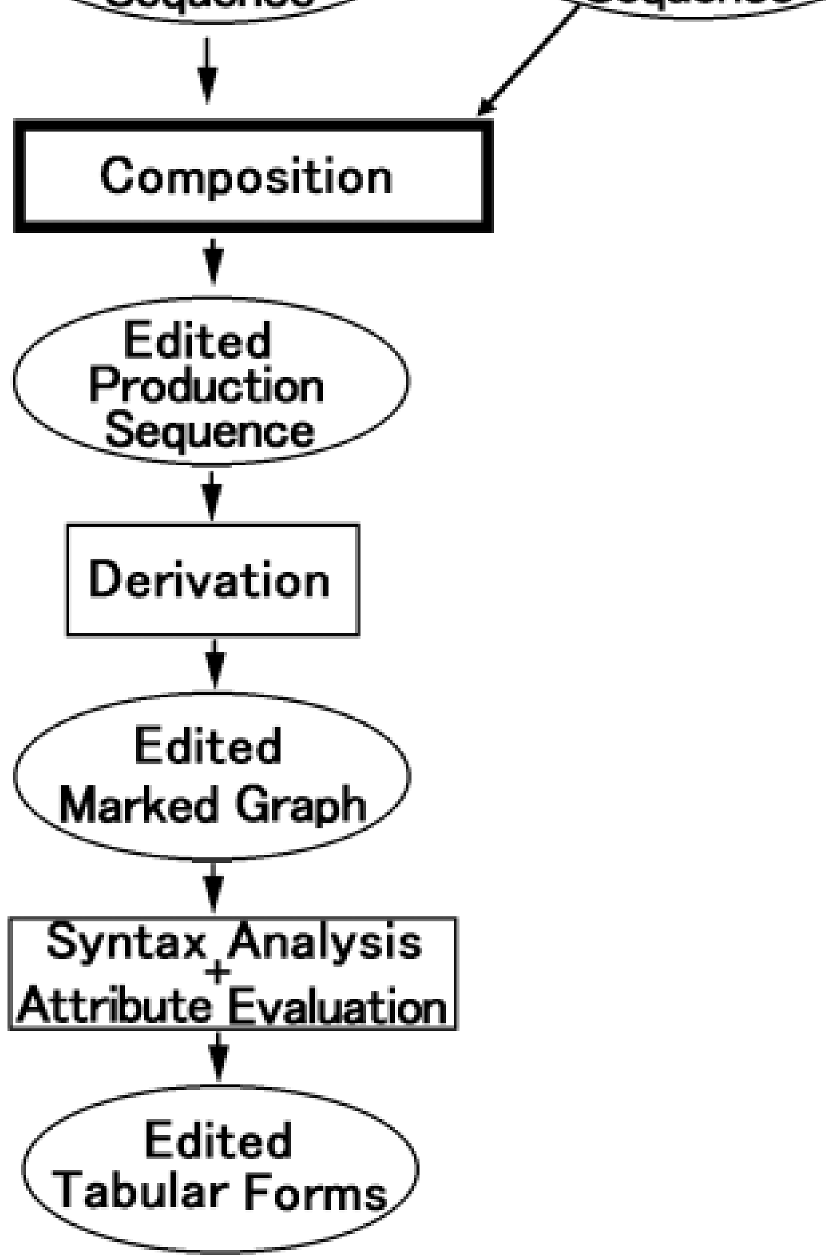
3.2.5 Proposition

Insertion in HNGG is executed in linear time.

□

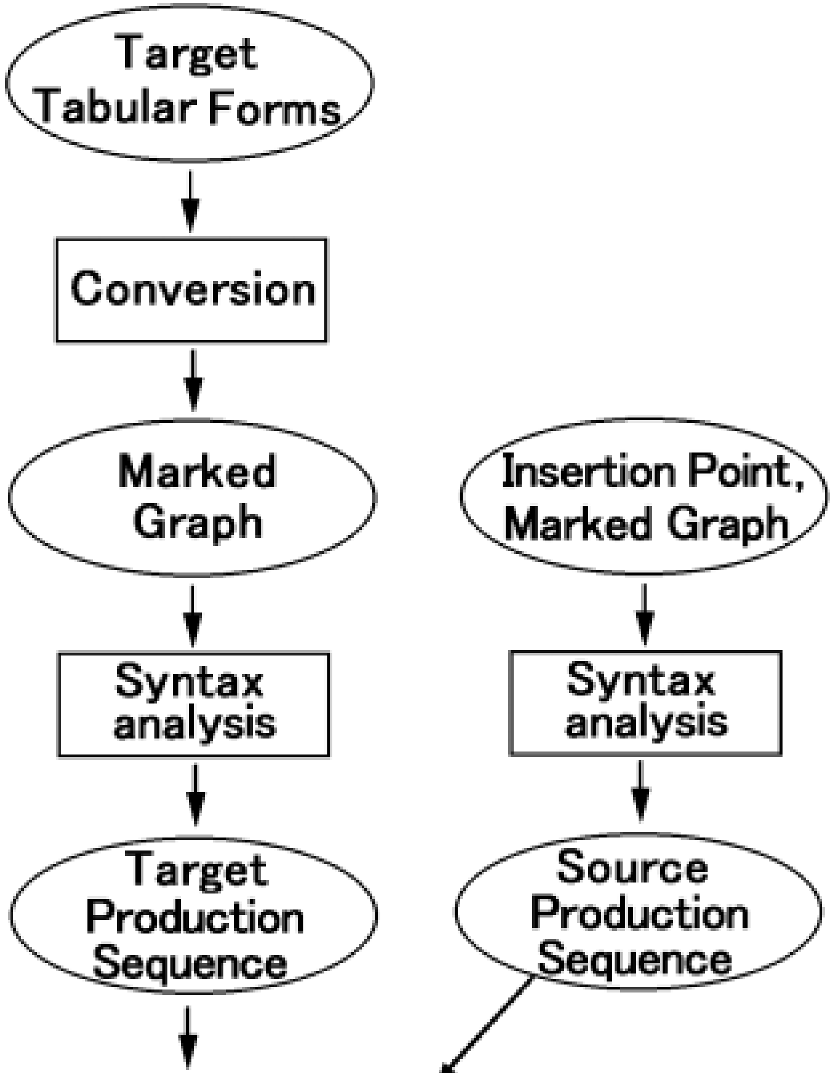
Derivation Tree of HNGG



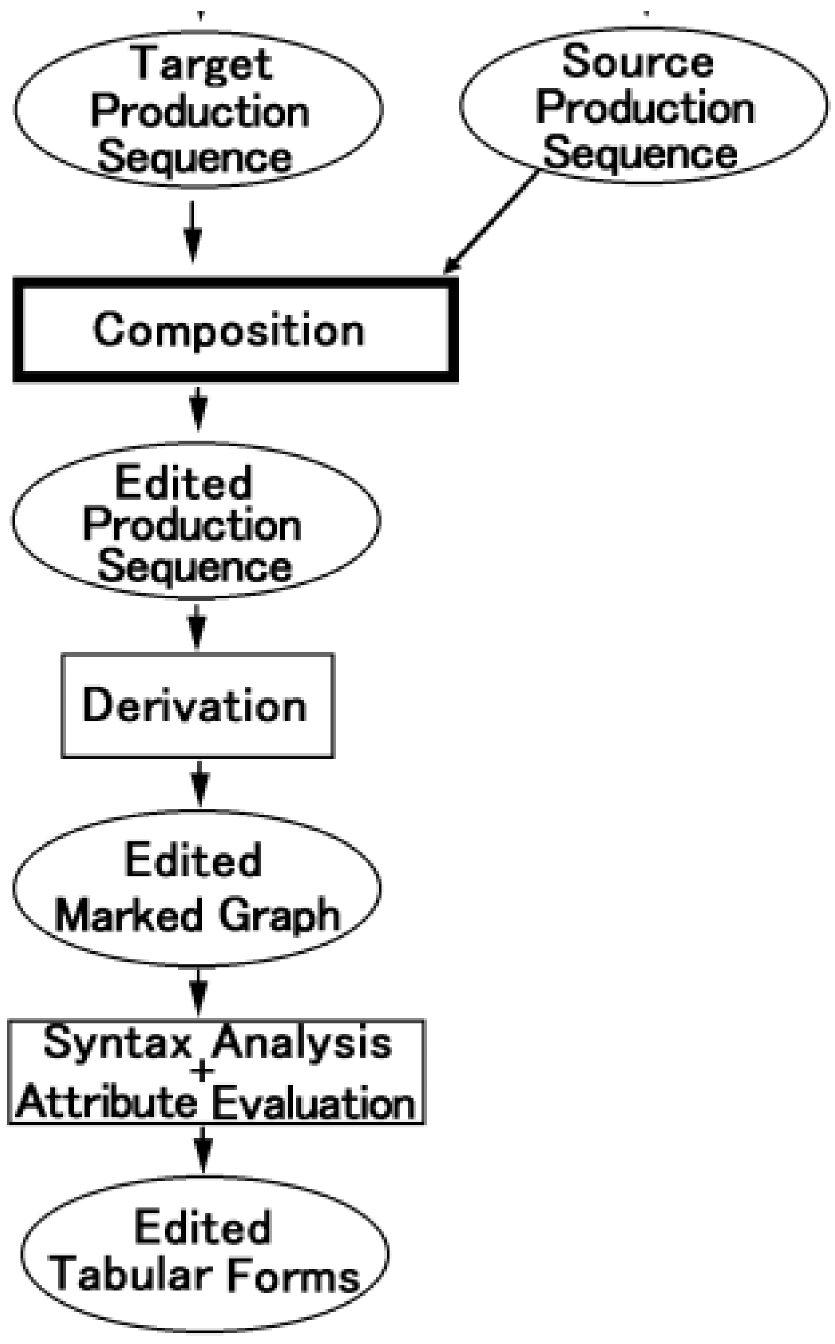


A process flow for an insertion of Hiform editing system

- Insertion process of editor



- Insertion process of editor (continued)



3.3 Syntactic Deletion of Item

3.3 Syntactic Deletion of Item

3.3.1 Definition (deletable)

For the derivation sequence $D_0 \overset{\omega 1\, H'p1}{\underset{p1}{\Rightarrow}} \cdots \overset{\omega k\, H'k}{\underset{pk}{\Rightarrow}} F \overset{\omega\, H'p}{\underset{p}{\Rightarrow}} D_p \overset{\omega l\, H'l}{\underset{pl}{\Rightarrow}} \cdots \overset{\omega n\, H'pn}{\underset{pn}{\Rightarrow}} D_n$,

The graph that D_p has node $u \in V_D$ for the first time

Node u is not applied to any production after that.

Production $p = X_p \rightarrow (D_p, C_p) \in P_N$ is <u>deletable</u> if one of the following Assumptions 1-3 is met

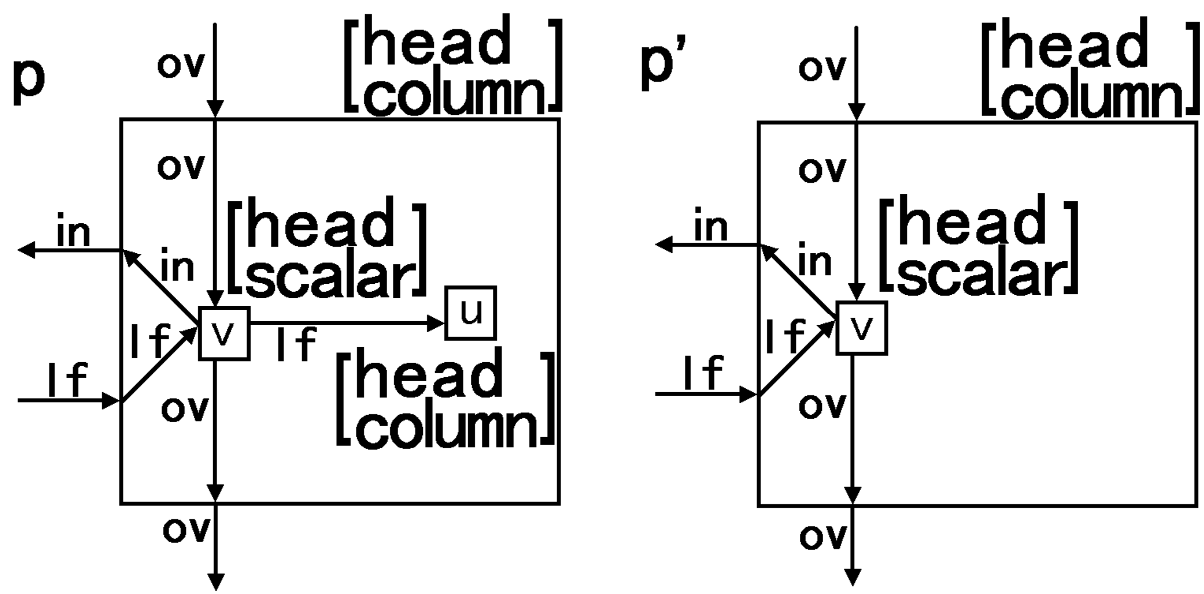
Assumption 1

For $p \in P_N$, $\exists$ $p': X_p \rightarrow (D_p, C_p) \in P_N$ s.t.

1. $X_{p'} = X_p$
2. $H_{p'} \equiv H'_p - \{u\}$
3. f、g : isomorphic mappings, $f: V_{H'p} \rightarrow V_{Hp}$ 、 $g: V_{Hp-\{f(u)\}} \rightarrow V_{Hp'}$

   then $(\sigma, \beta / \gamma, y, d) = (\sigma, \beta / \gamma, g(y), d)$

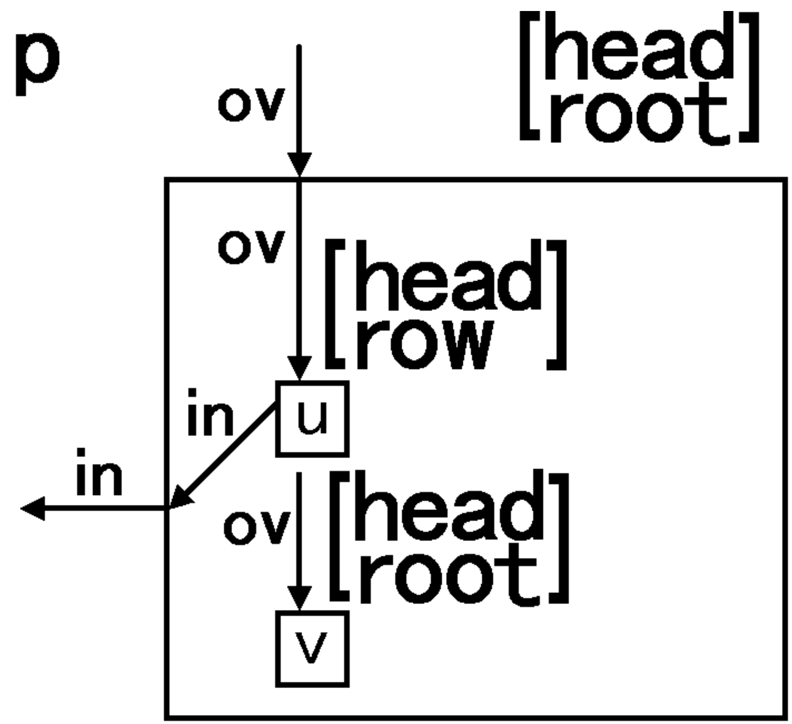
For example:



Assumption 2

$$V_{H'p} = \{u,v\}\ ,\quad X_{H'p} = \rho_{H'p}(v)$$

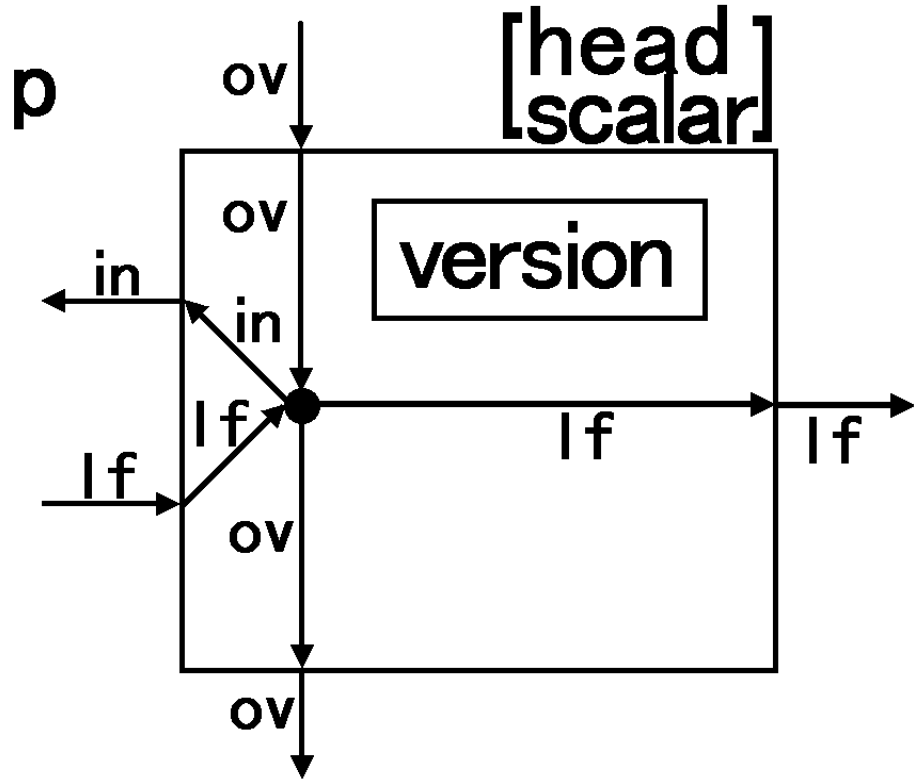
For example:



Assumption 3

$$\omega_j \notin H'_p \ , \quad 1 \leqq j \leqq n$$

For example:



□

Definition 3.3.2

A graph H' is obtained by <u>syntactic deletion</u> of a node A from a graph H

$\overset{def}{\Leftrightarrow}$

①A production q having a node A on the right hand side exists.

②The production q is deletable in the instance sequence for graph H.

③ $H \rightarrow \cdots \rightarrow H'$:

H' is the deleted graph which deletes the node A in the graph H.

3.2.4 Proposition

Let H be the graph obtained from G by the deletion of nodes a and b in this order, in HNGG.

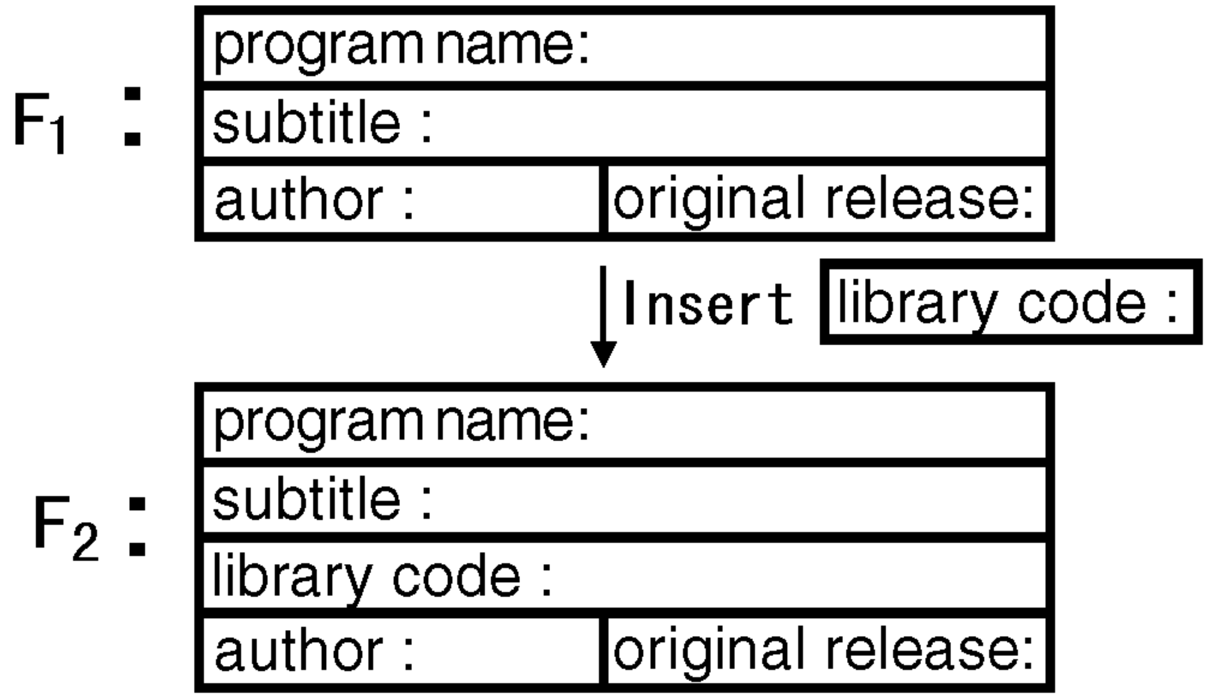
H' be the graph obtained from G by the deletion of nodes b and a in this order, in HNGG.

H=H'.

Proof. HNGG has confluence property. Thus, the theorem is verified. □

4 Example: Insertion Process

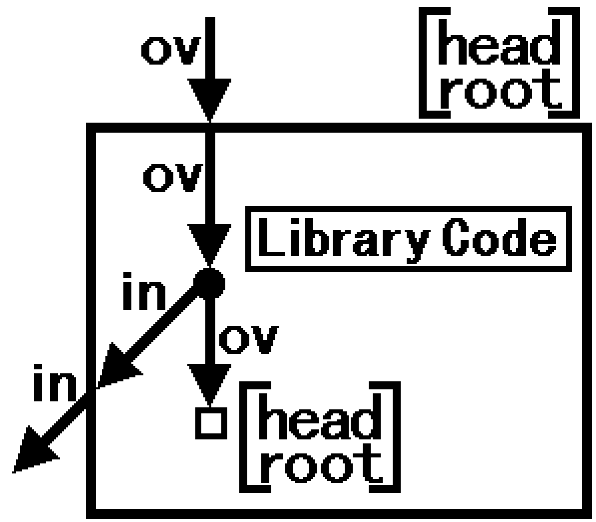
Insertion of library code between the 2nd and the 3rd branches of F1



Step1. It makes the composite production copy for the use of the insertion.

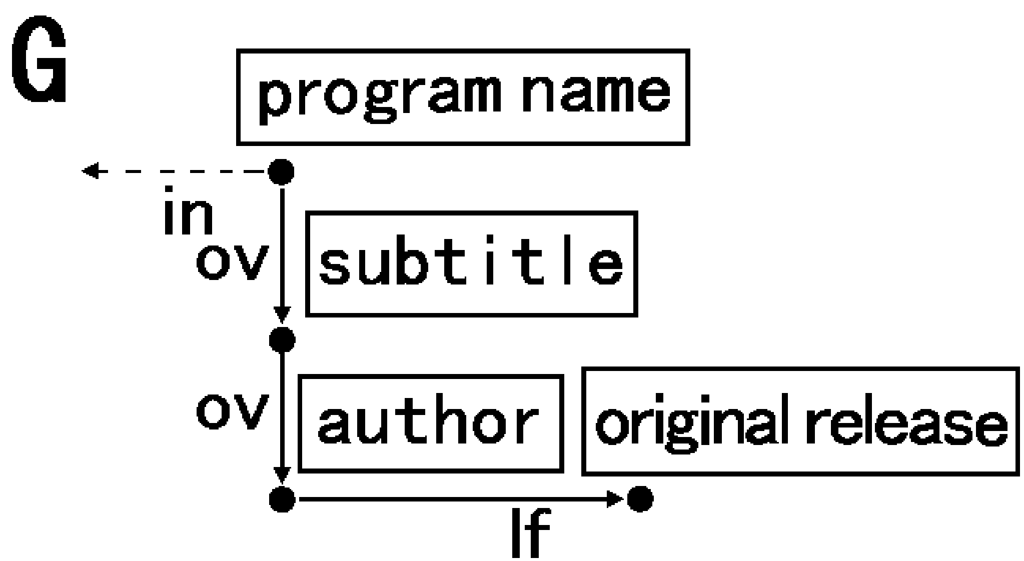
$$P_Q = ((((P_{H2} \circ P_{H4}) \circ P_{H5}) \circ P_{H6}) \circ P_{H9}) \circ P_{H10}$$

P_Q:



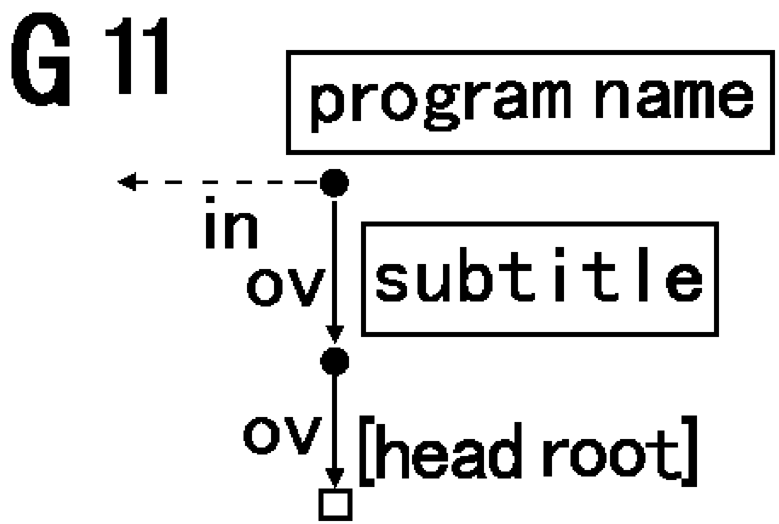
Step2. Construction of the Derivation D1 of graph G for form F1.

$D_1 : G_0 \underset{p1}{\overset{\omega 1H\text{þ}1}{\Rightarrow}} G_1 \underset{p2}{\overset{\omega 2H\text{þ}2}{\Rightarrow}} G_2 \underset{pH1}{\overset{\omega H1H\text{þ}H1}{\Rightarrow}} G_3 \underset{pH2}{\overset{\omega H2H\text{þ}H2}{\Rightarrow}} G_4 \underset{pH4}{\overset{\omega H4H\text{þ}H4}{\Rightarrow}} G_5 \underset{pH6}{\overset{\omega H6H\text{þ}H6}{\Rightarrow}} G_6 \underset{pH7}{\overset{\omega H7H\text{þ}H7}{\Rightarrow}} G_7 \underset{pH2}{\overset{\omega H2H\text{þ}H2}{\Rightarrow}} G_8 \underset{pH4}{\overset{\omega H4H\text{þ}H4}{\Rightarrow}}$
$G_9 \underset{pH6}{\overset{\omega H6H\text{þ}H6}{\Rightarrow}} G_{10} \underset{pH8}{\overset{\omega H8H\text{þ}H8}{\Rightarrow}} G_{11} \underset{pH3}{\overset{\omega H3H\text{þ}H3}{\Rightarrow}} G_{12} \underset{pH4}{\overset{\omega H4H\text{þ}H4}{\Rightarrow}} G_{13} \underset{pH5}{\overset{\omega H5H\text{þ}H5}{\Rightarrow}} G_{14} \underset{pH11}{\overset{\omega H11H\text{þ}H11}{\Rightarrow}} G_{15} \underset{pH6}{\overset{\omega H6H\text{þ}H6}{\Rightarrow}} G_{16} \underset{pH13}{\overset{\omega H13H\text{þ}H13}{\Rightarrow}}$
G



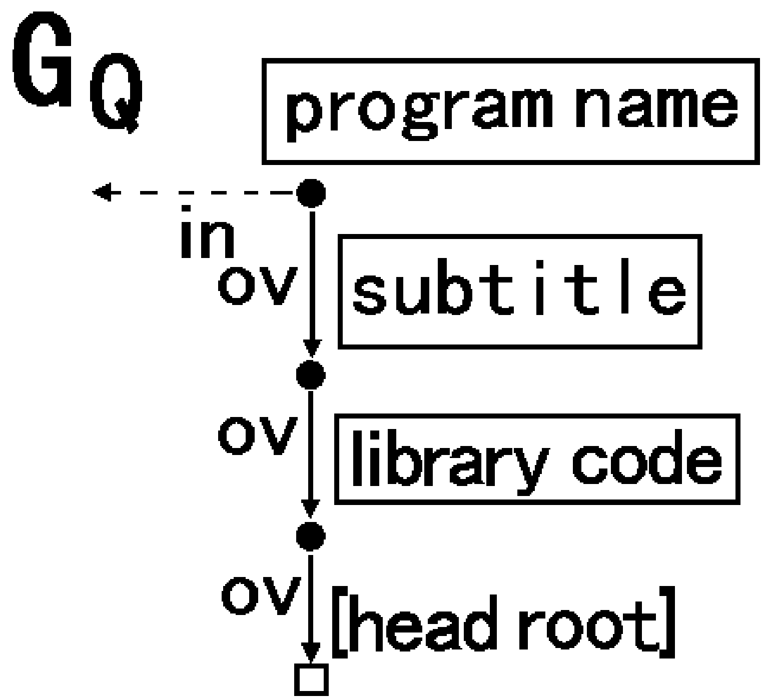
Step3. Find a target graph G11 by rows 2 and 3 in G.
Find a sub derivation D11 to generate G11 in D1.

$$D_{11} \quad : G_0 \underset{p1}{\overset{\omega 1\ H'p1}{\Rightarrow}} G_1 \underset{p2}{\overset{\omega 2\ H'p2}{\Rightarrow}} G_2 \underset{pH1}{\overset{\omega H1\ H'pH1}{\Rightarrow}} G_3 \underset{pH2}{\overset{\omega H2\ H'pH2}{\Rightarrow}} G_4 \underset{pH4}{\overset{\omega H4\ H'pH4}{\Rightarrow}} G_5 \underset{pH6}{\overset{\omega H6\ H'pH6}{\Rightarrow}} G_6 \underset{pH7}{\overset{\omega H7\ H'pH7}{\Rightarrow}} G_7 \underset{pH2}{\overset{\omega H2\ H'pH2}{\Rightarrow}} G_8$$
$$\underset{pH4}{\overset{\omega H4\ H'pH4}{\Rightarrow}} G_9 \underset{pH6}{\overset{\omega H6\ H'pH6}{\Rightarrow}} G_{10} \underset{pH8}{\overset{\omega H8\ H'pH8}{\Rightarrow}} G_{11}$$



Step4. Apply PQ to G11 and obtain GQ.

$$G_{11} \overset{\omega QH'pQ}{\underset{pQ}{\Rightarrow}} G_Q$$

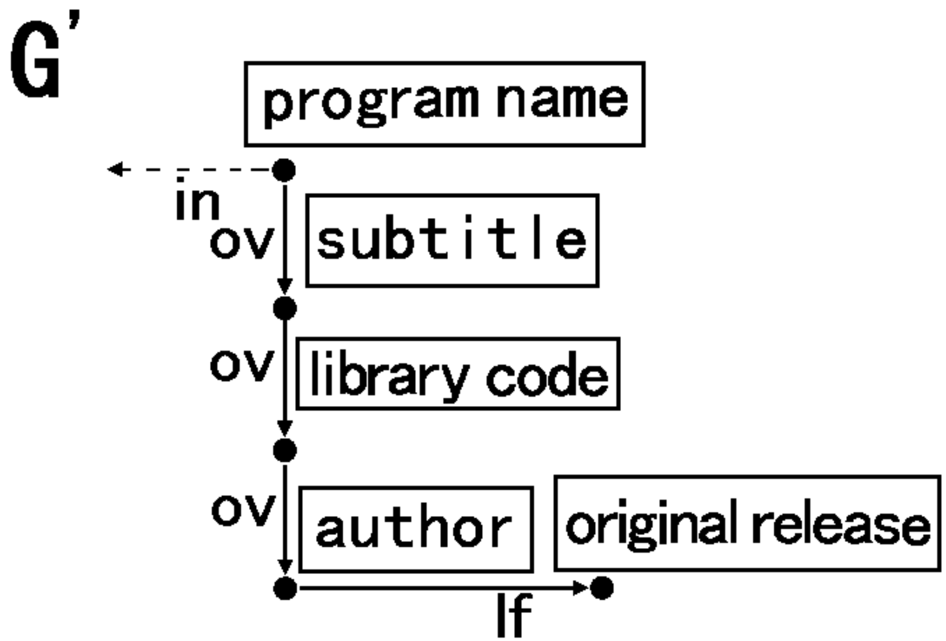


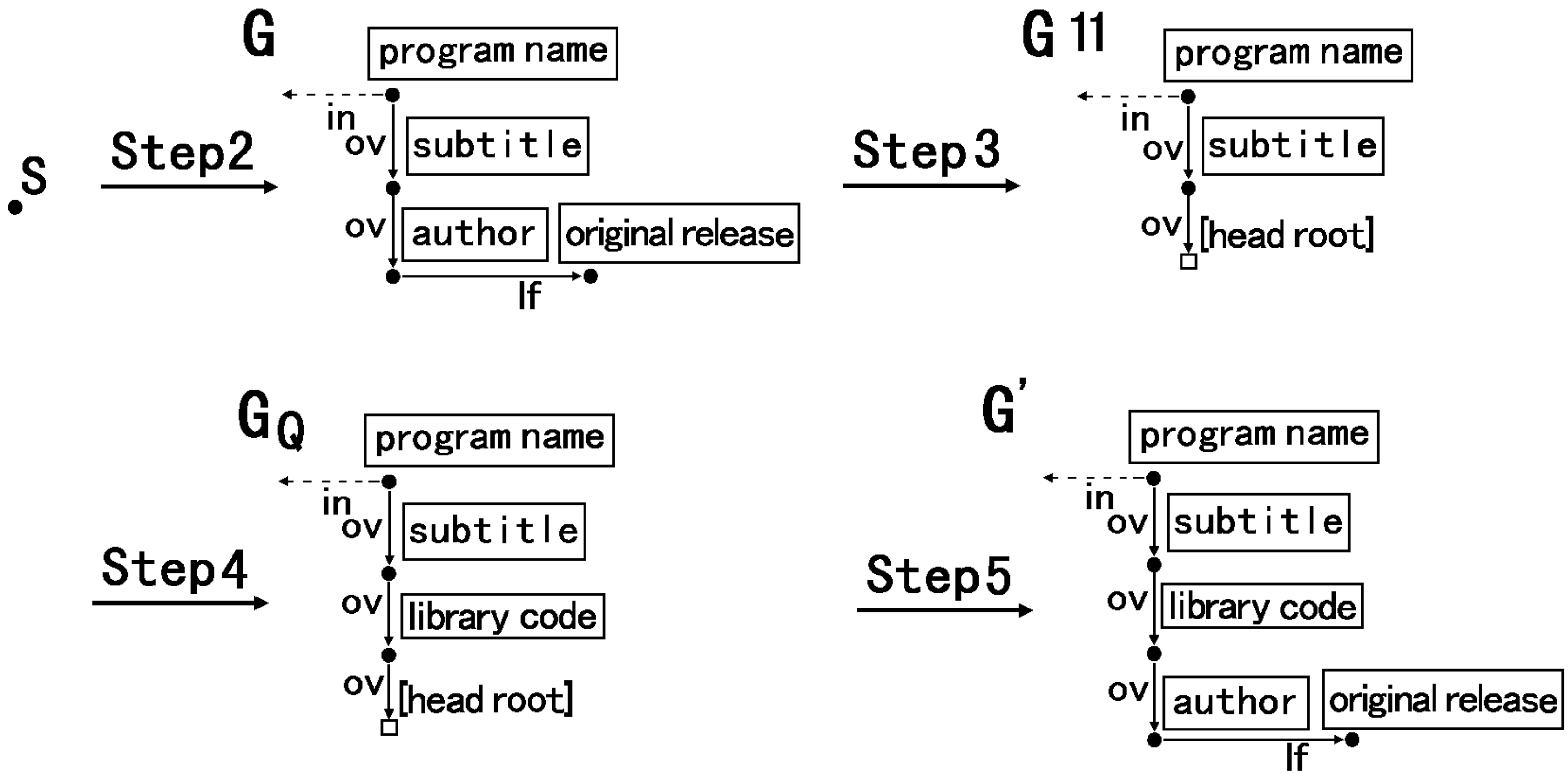
Step5. Apply the latter part of G23 to GQ.

$$G_Q \overset{\omega H2\,H'pH2}{\underset{pH2}{\Rightarrow}} G'_{12} \overset{\omega H4\,H'pH4}{\underset{pH4}{\Rightarrow}} G'_{13} \overset{\omega H5\,H'pH5}{\underset{pH5}{\Rightarrow}} G'_{14} \overset{\omega H11\,H'pH11}{\underset{pH11}{\Rightarrow}} G'_{15} \overset{\omega H6\,H'pH6}{\underset{pH6}{\Rightarrow}} G'_{16} \overset{\omega H13\,H'pH13}{\underset{pH13}{\Rightarrow}} G'_{17}$$

$$\overset{\omega H3\,H'pH3}{\underset{pH3}{\Rightarrow}} G'_{18} \overset{\omega H4\,H'pH4}{\underset{pH4}{\Rightarrow}} G'_{19} \overset{\omega H5\,H'pH5}{\underset{pH5}{\Rightarrow}} G'_{20} \overset{\omega H12\,H'pH12}{\underset{pH12}{\Rightarrow}} G'_{21} \overset{\omega H6\,H'pH6}{\underset{pH6}{\Rightarrow}} G'_{22} \overset{\omega H14\,H'pH14}{\underset{pH14}{\Rightarrow}} G'$$

G' is a graph for form F_2.



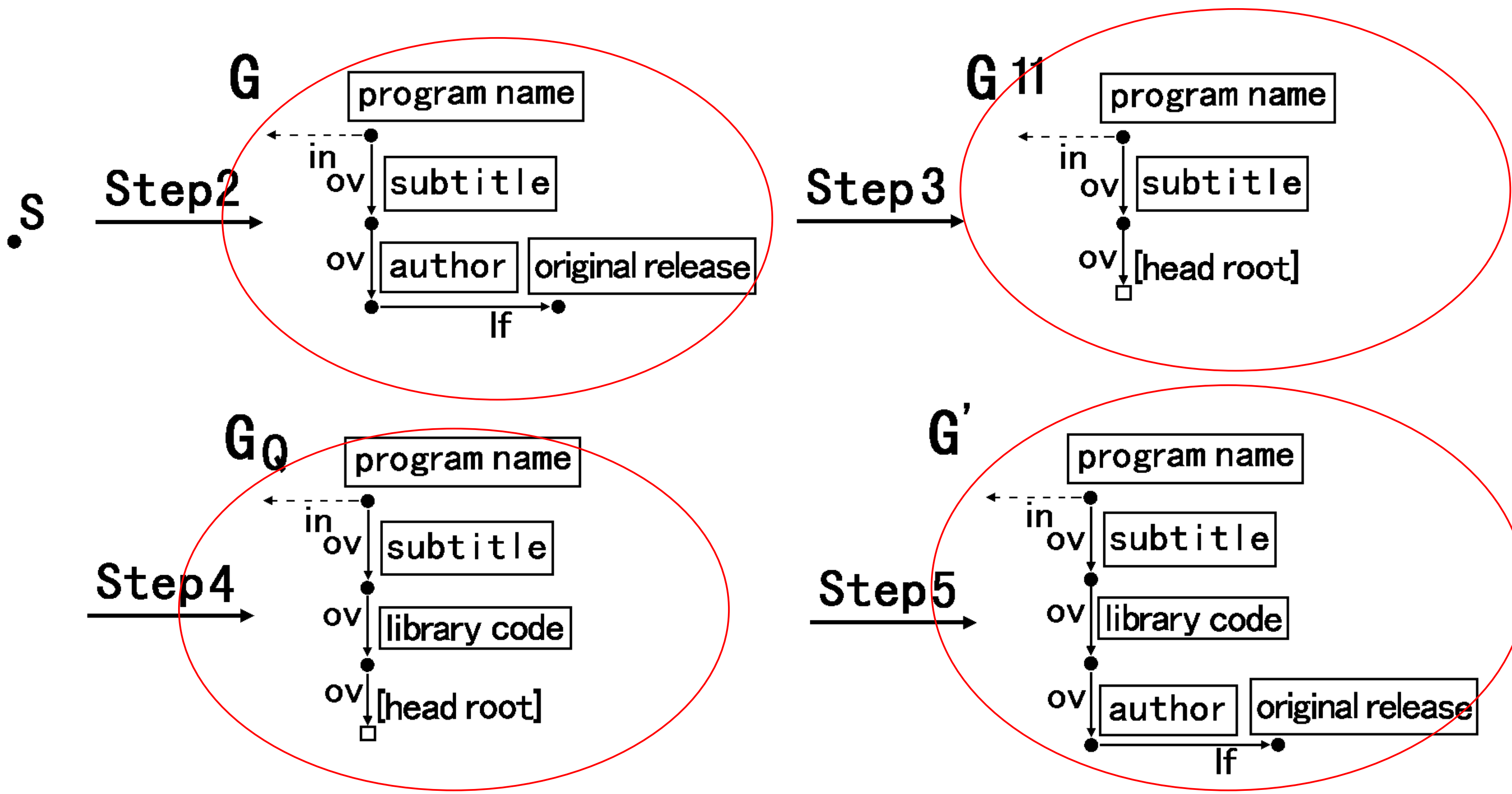


Insertion process

$$D_1 : G_0 \xRightarrow[p1]{\omega 1\ H'p1} G_1 \xRightarrow[p2]{\omega 2\ H'p2} G_2 \xRightarrow[pH1]{\omega H1\ H'pH1} G_3 \xRightarrow[pH2]{\omega H2\ H'pH2} G_4 \xRightarrow[pH4]{\omega H4\ H'pH4} G_5 \xRightarrow[pH6]{\omega H6\ H'pH6} G_6 \xRightarrow[pH7]{\omega H7\ H'pH7} G_7$$

$$\xRightarrow[pH2]{\omega H2\ H'pH2} G_8 \xRightarrow[pH4]{\omega H4\ H'pH4} G_9 \xRightarrow[pH6]{\omega H6\ H'pH6} G_{10} \xRightarrow[pH8]{\omega H8\ H'pH8} G_{11} \xRightarrow[pQ]{\omega Q\ H'pQ} G_Q \xRightarrow[pH3]{\omega H3\ H'pH3} G_{12} \xRightarrow[pH4]{\omega H4\ H'pH4} G_{13}$$

$$\xRightarrow[pH5]{\omega H5\ H'pH5} G_{14} \xRightarrow[pH11]{\omega H11\ H'pH11} G_{15} \xRightarrow[pH6]{\omega H6\ H'pH6} G_{16} \xRightarrow[pH13]{\omega H13\ H'pH13} G$$



Insertion process

5.Conclusion

- We proposed syntactic editing methods for tabular forms, based on the attribute edNCE graph grammar.
  - Insert manipulation (Section 3.2)
  - Delete manipulation (Section 3.3)
- Linear time editing algorithm with attribute rules for primitive drawing

Future Works

- Incremental Algorithm
- Attribute Rules for more sophisticated drawing
- Other edit manipulations representing a division manipulation, a combination manipulation and so on.
- We are now developing a tabular form editor system utilizing this approach.