

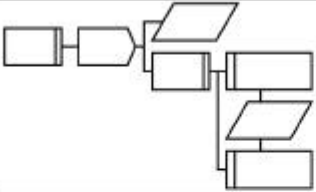
Syntactic Processing of Diagrams by Graph Grammars

Tomokazu ARITA, Kiyonobu TOMIYAMA, Takeo YAKU
(Nihon Univ.)

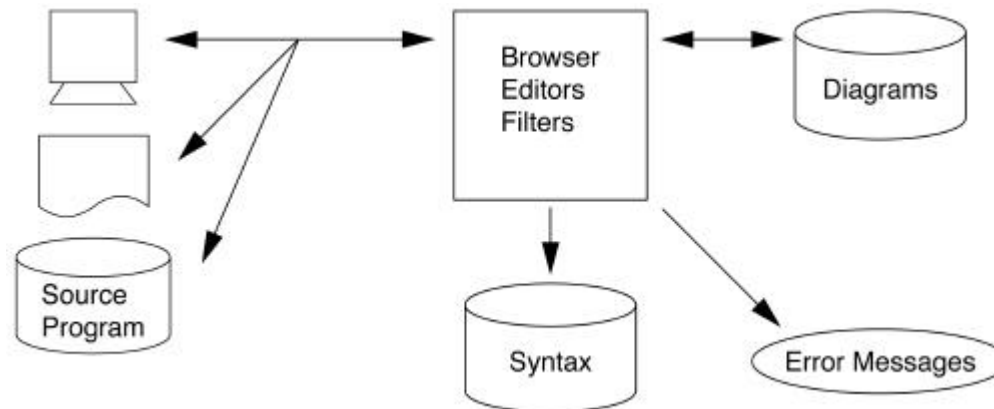
Youzou MIYADERA, Kimio SUGITA, Kensei TSUCHIDA
(Tokyo Gakugei Univ.) (Tokai Univ.) (Toyo Univ.)

Abstract

■ TARGET: Diagrams in Software Specification

	Diagrams	Corresponding Graphs	Universal Models												
Hierarchical Diagram		Attribute Tree	Attribute NCE CFGG												
Nested Diagram	<table border="1" data-bbox="667 683 1126 770"> <tr> <td>Program Code:</td> <td rowspan="3">Program Specification</td> </tr> <tr> <td>Program Name:</td> </tr> <tr> <td>Library Code:</td> <td>Version:</td> </tr> </table>	Program Code:	Program Specification	Program Name:	Library Code:	Version:	Attribute Marked Tree	Attribute NCE CFGG							
Program Code:	Program Specification														
Program Name:															
Library Code:		Version:													
Tessellation Diagram	<table border="1" data-bbox="667 826 1126 914"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Size</th> <th>G/L</th> </tr> </thead> <tbody> <tr> <td>x</td> <td>int</td> <td>2</td> <td>G</td> </tr> <tr> <td>y</td> <td>float</td> <td>4</td> <td>L</td> </tr> </tbody> </table>	Name	Type	Size	G/L	x	int	2	G	y	float	4	L	Attribute Marked Tessellation Graph	Attribute NCE CSGG
Name	Type	Size	G/L												
x	int	2	G												
y	float	4	L												

■ GOAL: Syntactic Processing



Abstract (continued)

- **PARADIGM** : Attribute Graph Grammars

- Rewriting rules for syntax
- Attribute rules for graph drawing

- **Our Solution**

- **Attribute NCE Graph Grammars**

Diagram	Rewriting Rules	Attribute Rules
Hierarchical Diagram	67	723
Nested Diagram	280	1248
Tessellation Diagram	69	308

- **Syntactic Processing Methods of Diagrams**

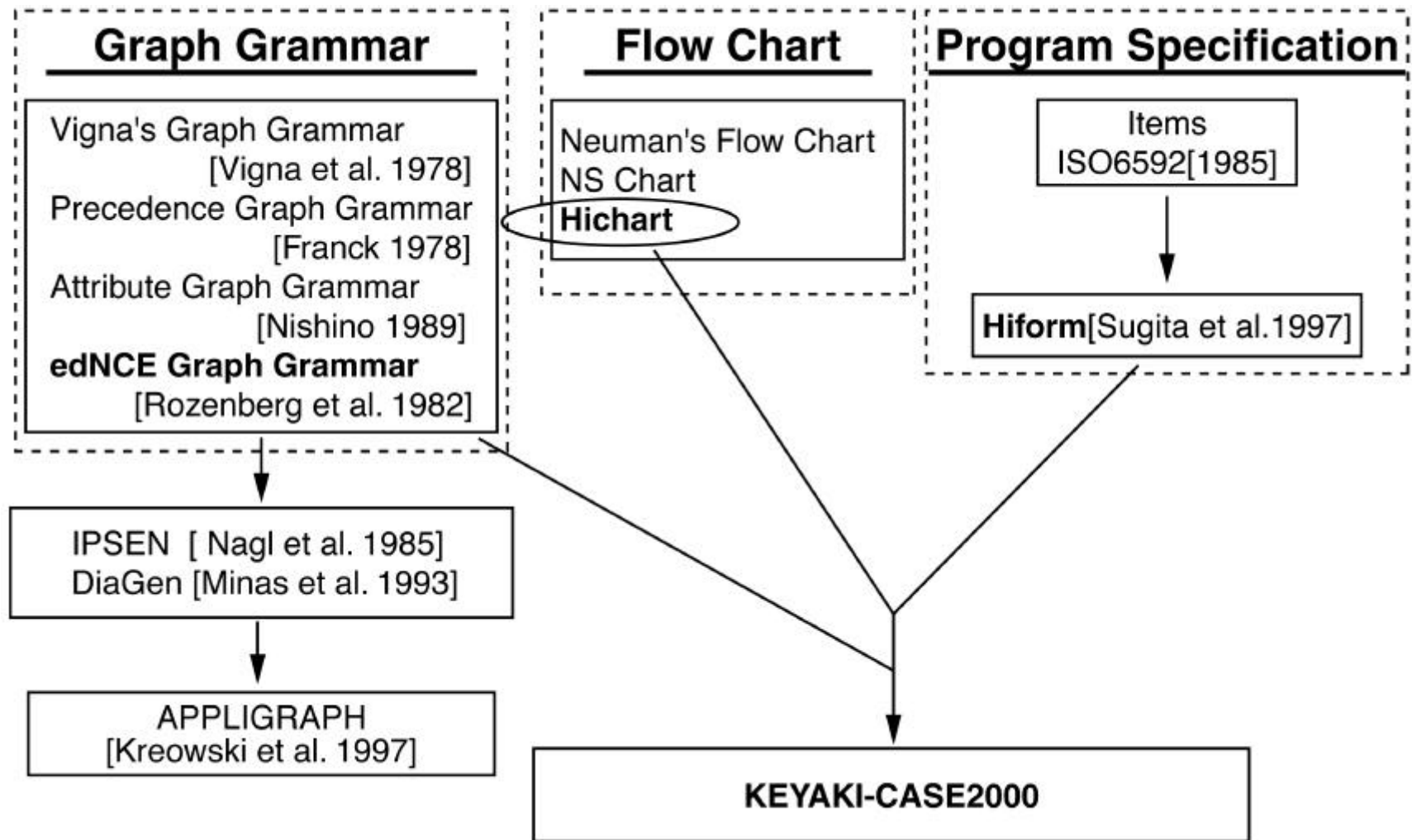


Contents

- 1. Introduction
- 2. Program Flowcharts and Specification Forms
- 3. An Attribute Graph Grammar for Hierarchical Diagrams
- 4. Attribute Graph Grammars for Tabular Diagrams
- 5. Diagram Processing System
- 6. Conclusion

1 Introduction

Background





Our History

1978 Graphical notation of program flowchart Hichart

1987 Hichart (non-syntactic)[COMPSAC 11]

1996 Hichart Diagram Editing Command [COMPSAC 21]
(based on Vigna's context-free graph grammar)

1998 CAI system [APEC-CIL'97]

1998 Program Visualization [IFIP98, ICSE98]

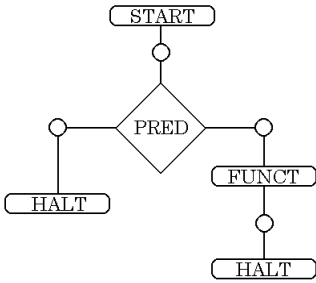
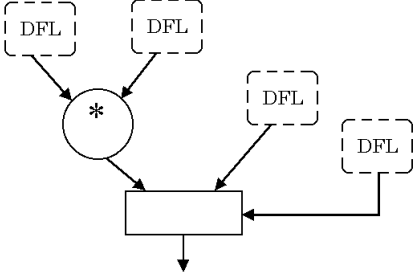
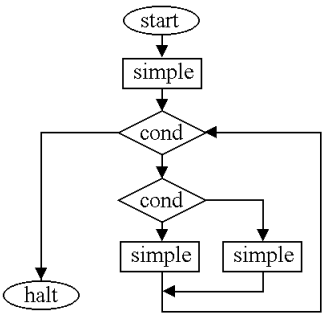
2000 Syntactic Processing of Diagrams by Graph Grammars[IFIP WCC 2000]



Related Works

Project Name	Program Semantics by GG	Drawing by Combinatorial Algorithm	Drawing by AGG	Syntax by Graph Grammar
IPSEN	○	—	—	○
DiaGen	—	○	—	○
KEYAKI-CASE2000	—	○	○	○

Related Works (continued)

Diagrams	Known Models	
<p data-bbox="259 368 539 424">Flowchart</p>  <pre> graph TD START([START]) --> PRED{PRED} PRED --> HALT1([HALT]) PRED --> FUNCT([FUNCT]) FUNCT --> HALT2([HALT]) </pre>	<p data-bbox="965 368 1238 501">CF PLEX Grammar</p>	<p data-bbox="1520 368 1917 424">K.S.Fu (1982)</p>
<p data-bbox="259 729 421 970">Data Flow Chart</p>  <pre> graph TD DFL1[DFL] --> M((*)) DFL2[DFL] --> M M --> B[] DFL3[DFL] --> B DFL4[DFL] --> B B --> Out[] </pre>	<p data-bbox="965 729 1245 861">Positional Grammar</p>	<p data-bbox="1520 729 1921 884">G. Costagliola et al. (1990)</p>
<p data-bbox="259 1074 555 1315">Structured Flow Chart</p>  <pre> graph TD start([start]) --> simple1[] simple1 --> cond1{cond} cond1 --> cond2{cond} cond2 --> simple2[] cond2 --> simple3[] simple2 --> cond1 simple3 --> cond1 cond1 --> halt([halt]) </pre>	<p data-bbox="965 1074 1433 1222">Symbol Relation Grammar</p>	<p data-bbox="1520 1074 1861 1228">F.Ferruci et al. (1996)</p>



Motivation

- Formalism of diagram's structure and layout information
- Formalism of diagram processing method

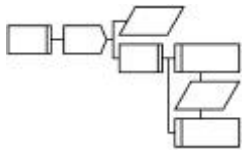


Purpose

- To *characterize* types of graph grammars that generate three types of diagrams
- To propose *integrated processing methods* of diagrams using the graph grammars

Results

■ Types of graph grammars

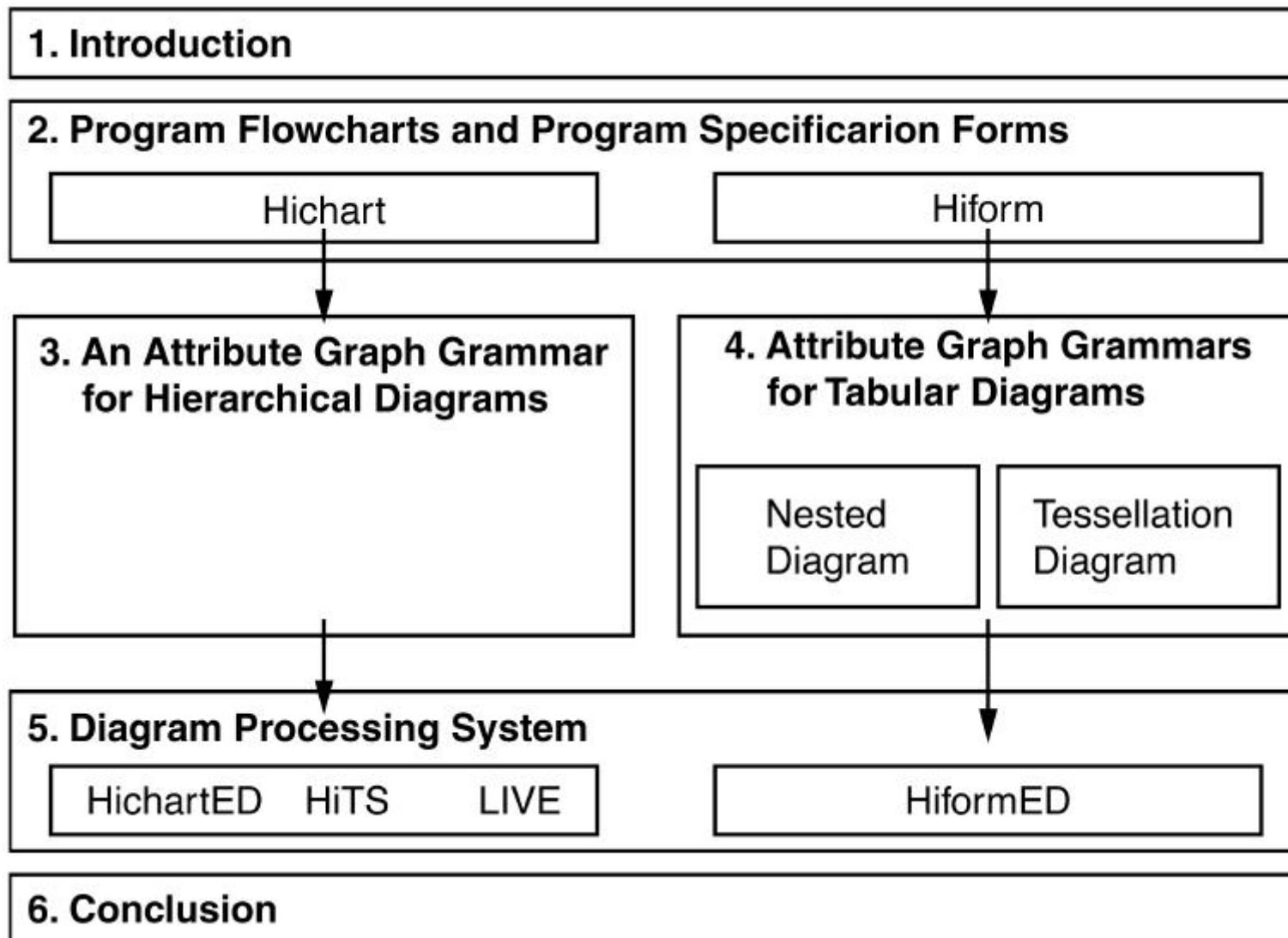
Graph Grammar	Grammar's type (Rewriting rule, Attribute rule)	Diagram												
HCGG	<i>Context-free</i> (67, 723)													
HNGG	<i>Context-free, precedence</i> (280, 1248)	<table border="1" data-bbox="1442 991 1906 1082"> <tr> <td>Program Code:</td> <td>Program Specification</td> </tr> <tr> <td>Program Name:</td> <td></td> </tr> <tr> <td>Library Code:</td> <td>Version:</td> </tr> </table>	Program Code:	Program Specification	Program Name:		Library Code:	Version:						
Program Code:	Program Specification													
Program Name:														
Library Code:	Version:													
HTGG	<i>Context-sensitive</i> (69, 308)	<table border="1" data-bbox="1442 1139 1899 1251"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Size</th> <th>G/L</th> </tr> </thead> <tbody> <tr> <td>x</td> <td>int</td> <td>2</td> <td>G</td> </tr> <tr> <td>y</td> <td>float</td> <td>4</td> <td>L</td> </tr> </tbody> </table>	Name	Type	Size	G/L	x	int	2	G	y	float	4	L
Name	Type	Size	G/L											
x	int	2	G											
y	float	4	L											

■ Integrated processing methods of diagrams

Diagram Processing System **KEYAKI-CASE2000**



Contents

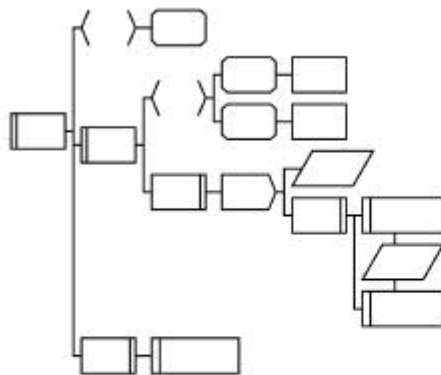


2. Program Flowcharts and Program Specification Forms

Diagram in Program Specifications

Hichart

Hierarchical Diagram



Hiform

(Tabular Diagrams)

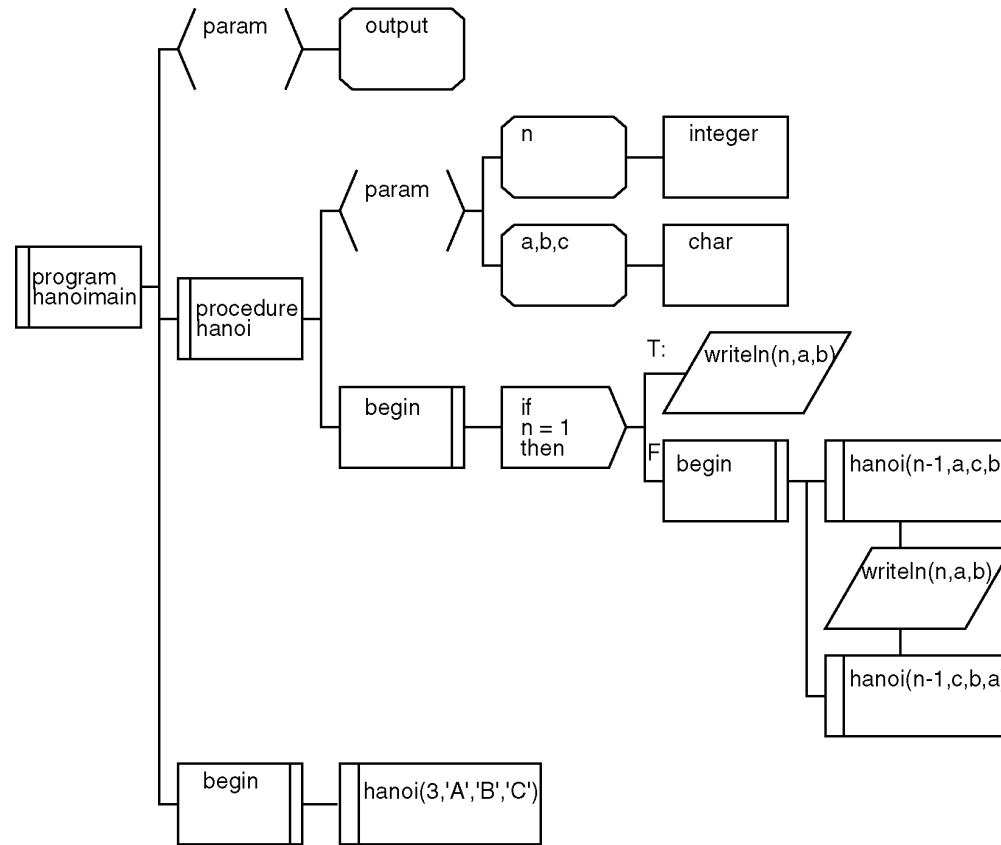
Nested Diagram

Program Code:	Program Specification
Program Name:	
Library Code:	Version:

Tessellation Diagram

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

2.1 Hierarchical Diagram for Program Flowchart



A Hichart program flowchart (Tower of Hanoi).

2.2 Tabular Diagrams for Program Specification Forms

Hiform

(a program specification language)

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

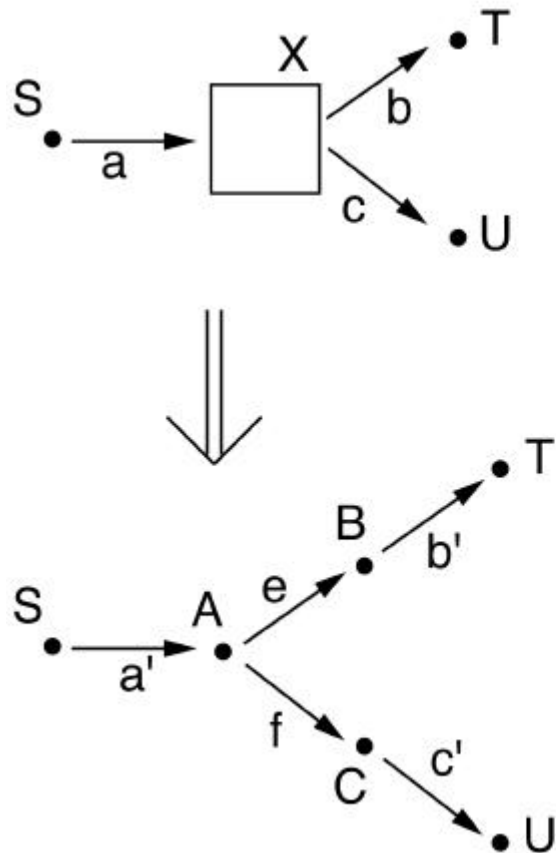
Project Code:	A 5
Program Name:	Program Specification-1 p
Library Code:	Version:
Author:	Original Release:
Approver:	Current Release:
Problem Description:	
Problem Supplementary Information (Theoretical Principles, Methods and References):	
Problem Solution: 1.Conventions and Terminology 2.Principles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



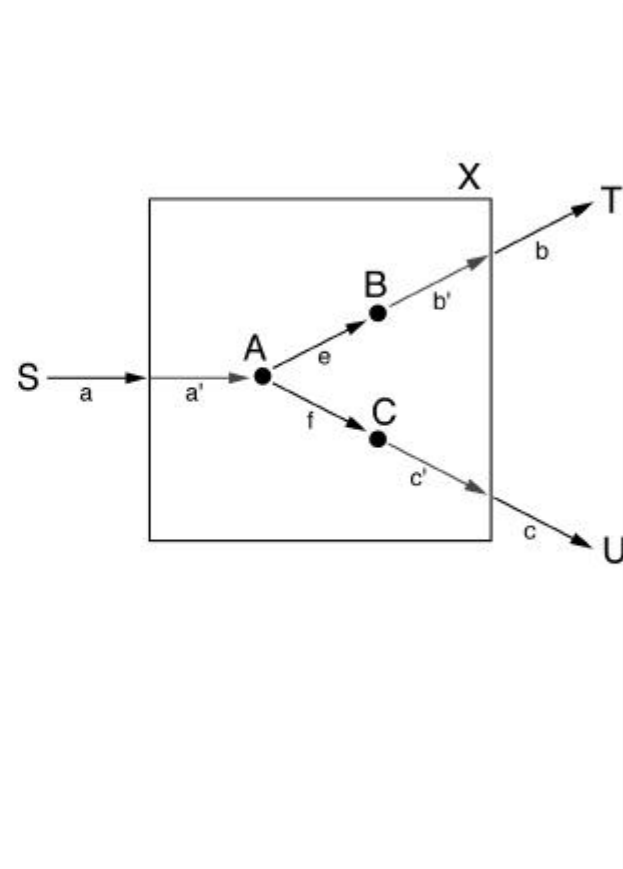
3. An Attribute Graph Grammar for Hierarchical Diagrams

Attribute Context-sensitive NCE (Neighborhood Control Embedding) Graph Grammar [Rozenberg et al. 1982]

Derivation :

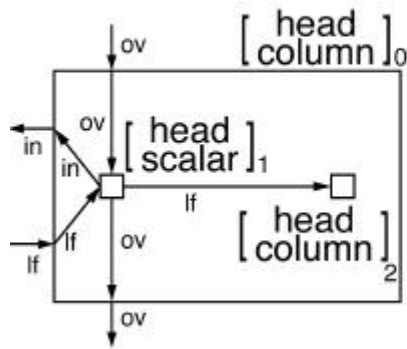


Rewriting Rule:



Attribute Context-free NCE Graph Grammar (continued)

Production with attribute rules:



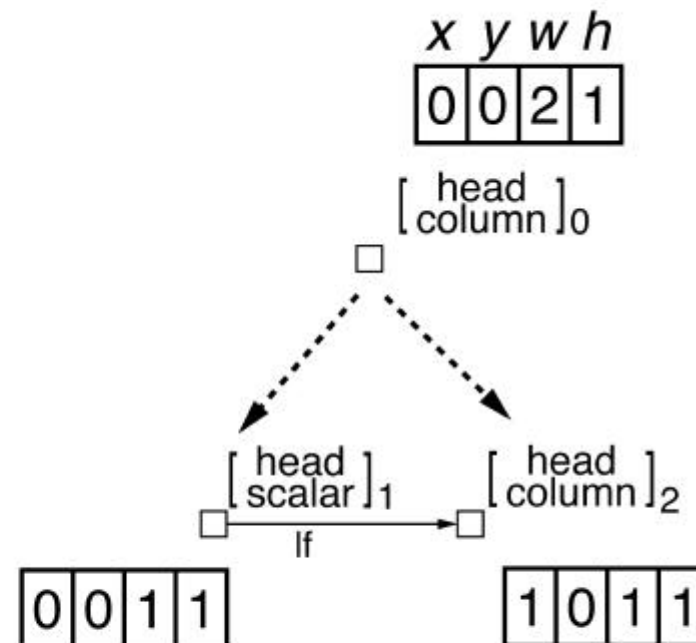
Semantic Rule

```

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))
    
```

Attribute rule evaluation in CF NCE GG :

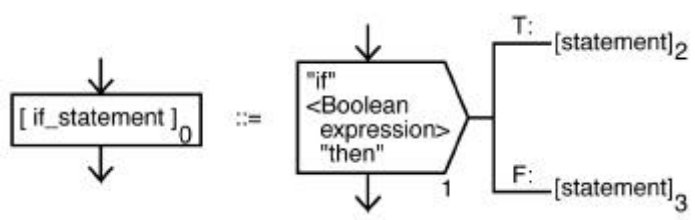


Remark: Our Attribute context-sensitive
NCE Graph Grammar rewrites exactly one
Node in each derivation [cf. Adachi-Yaku (1999)].

■ Grammar 3.1 HCGG

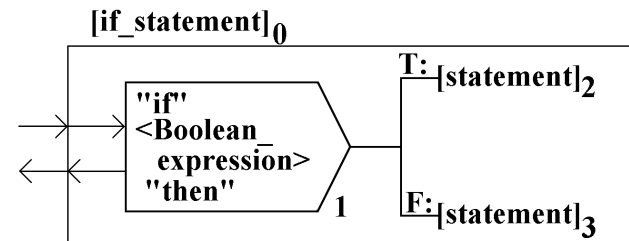
HCGG (HiChart Graph Grammar) is an attribute context-free NCE graph grammar for hierarchical diagrams in Hichart such as :

Production Example of HCGG



if_statement (1)

Production



Semantic Rules

$top(2)=top(0)$	$cl(2)="T:"$
$top(3)=bottom(2)+GapY$	$cl(3)="F:"$
$x(1)=x(0)$	$id(1)=id(0)$
$x(2)=x(0)+w(1)+GapX$	$id(2)=id(1)+1$
$x(3)=x(0)+w(1)+GapX$	$id(3)=id(2)+nc(2)$
$y(0)=(y(2)+y(3))/2$	$nc(0)=1+nc(2)+nc(3)$
$bottom(0)=max(bottom(1),bottom(3))$	

$w(1)=MinW$

$h(1)=get_height(["if",<Boolean_expression>,"then"])$

$cell(1)="exclusive_selection"$

$string(1)=get_str(["if",<Boolean_expression>,"then"])$

$lines(1)=get_line(1,[2,3])$

Features of HCGG

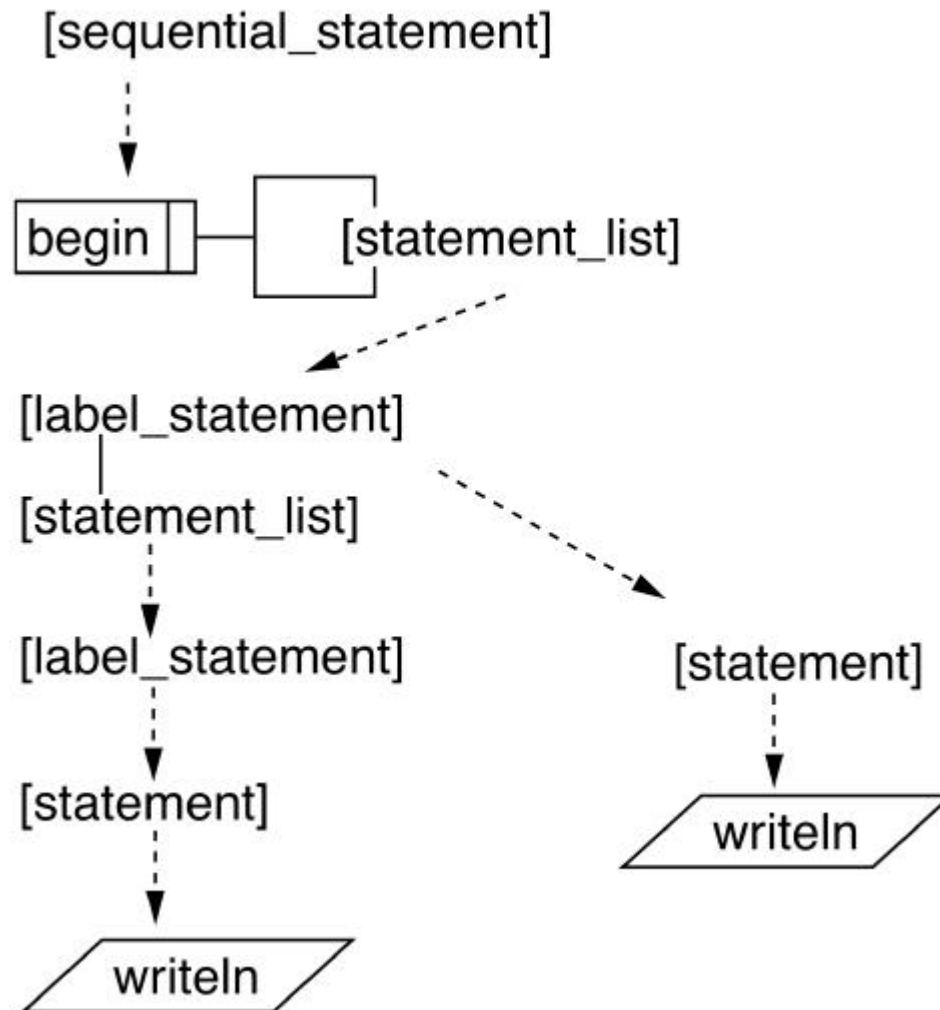
GG	Type	Rewriting Rule	Attribute Rule
HCGG	Context-free	67	723



■ Property 3.2

Attribute rules in HCGG are evaluated in linear time.

■ Derivation Tree of HCGG

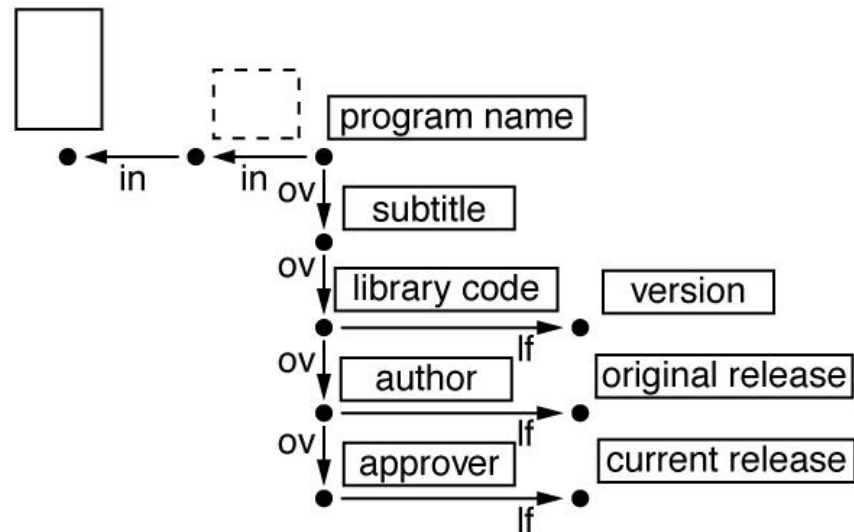


4. Attribute Graph Grammars for Tabular Diagrams

■ Nested Diagram and Its Corresponding Marked Graph

program name :	
subtitle :	
library code :	version :
author :	original release :
approver :	current release :

program name :	
subtitle :	
library code :	version :
author :	original release :
approver :	current release :

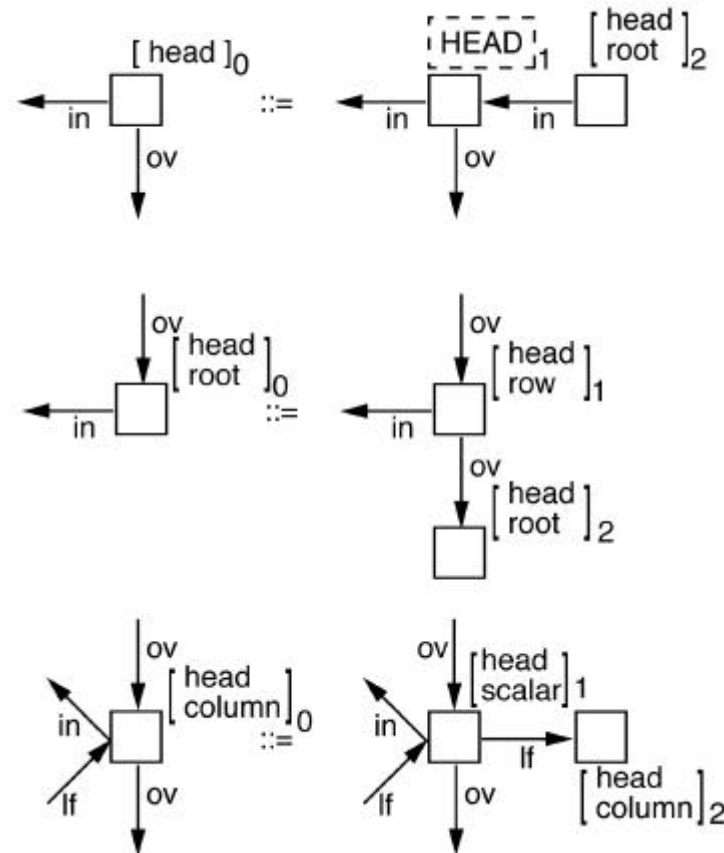
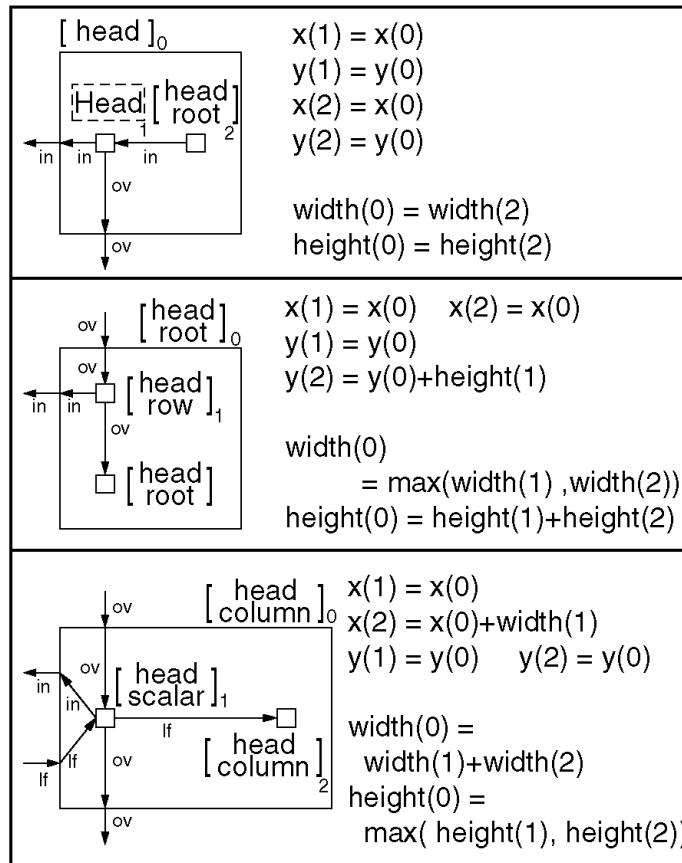


■ Grammar 4.1 HNGG

Nested Diagram

HNGG (Hiform Nested Graph Grammar) is an attribute context-free NCE graph grammar for the nested diagrams such as:

Production Examples of HNGG

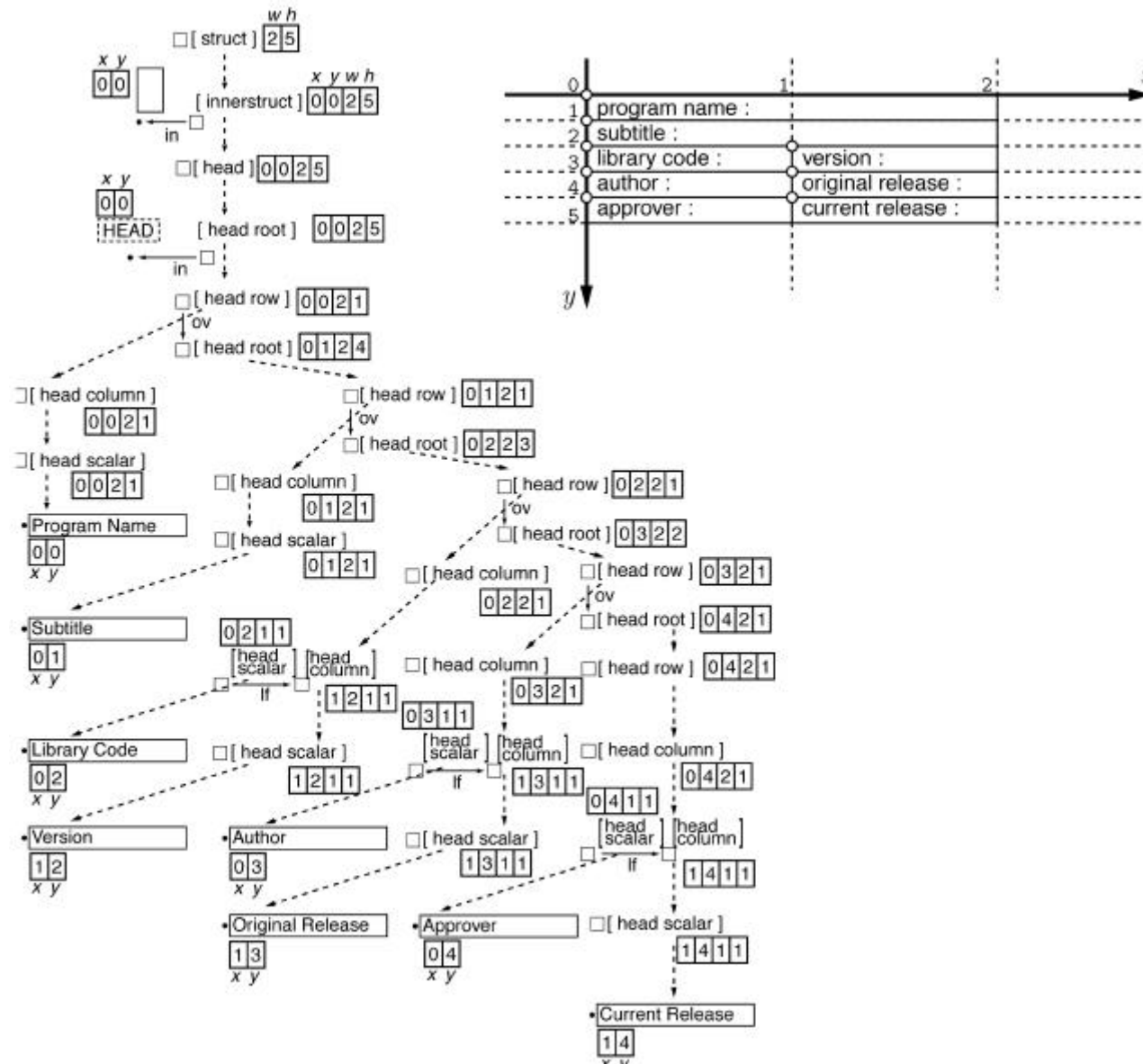


Features of HNGG

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



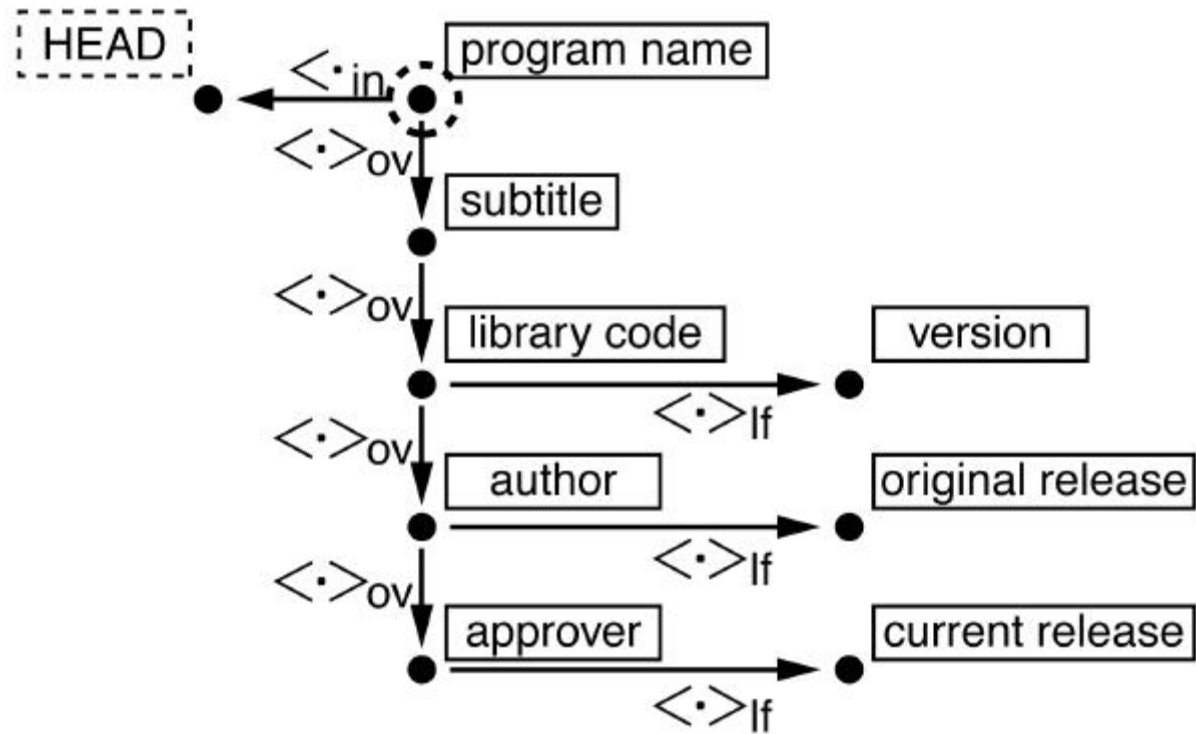
Derivation Tree of HNGG



■ **Property 4.2**

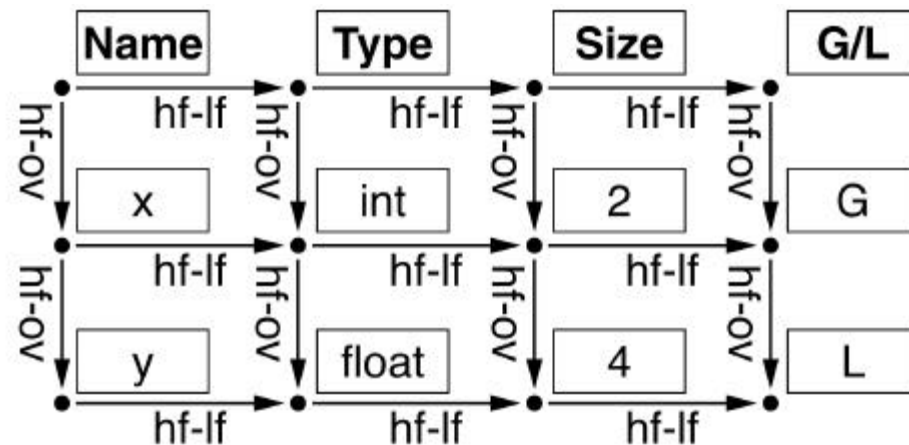
HNGG is a precedence graph grammar (see e.g. Franck 1978).

■ How to use precedence rule



■ Tessellation Diagram and Its Corresponding Graph

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

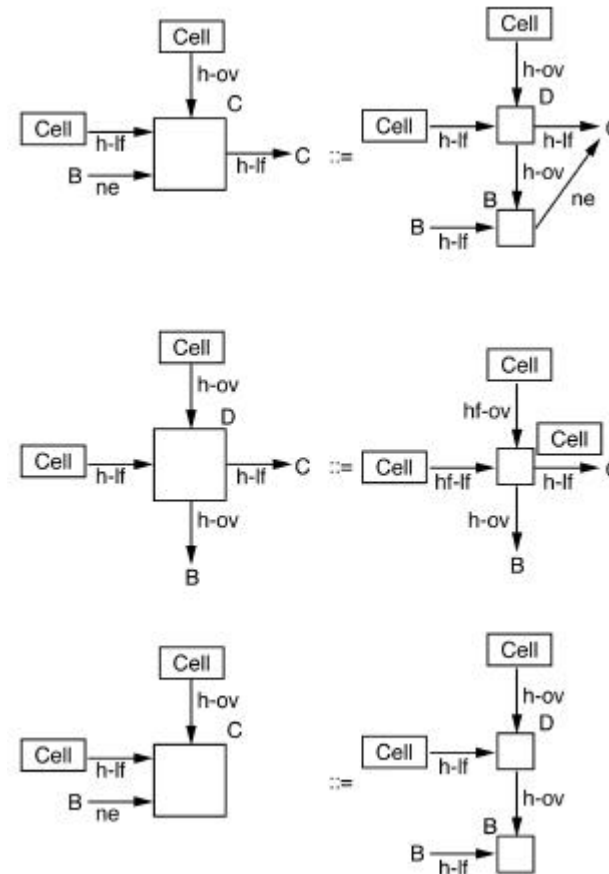
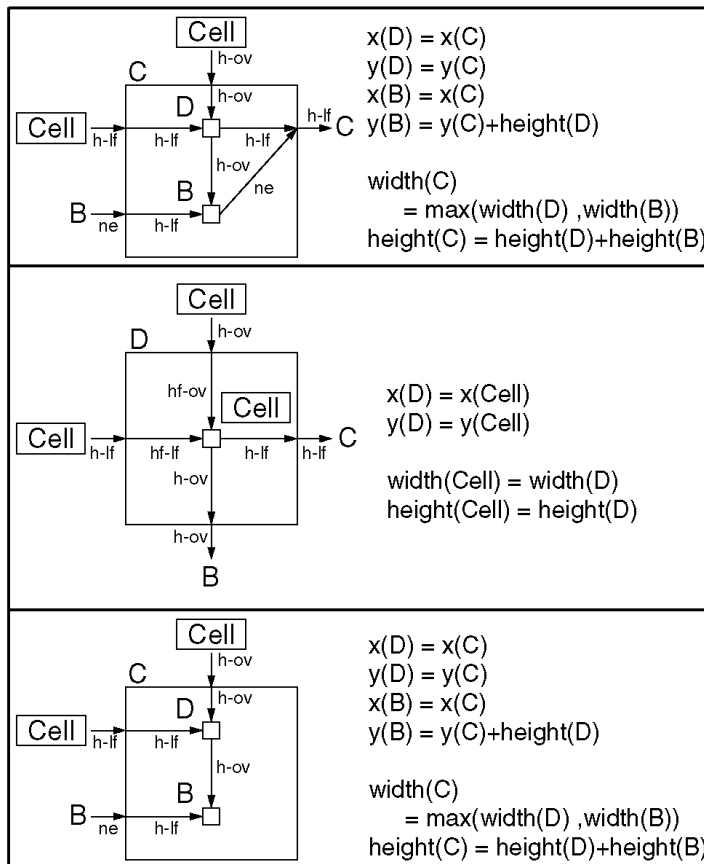


■ Grammar 4.3 HTGG

Tessellation Diagram

HTGG (Hiform Tessellation Graph Grammar)
is an attribute context-sensitive NCE graph grammar for the tessellation diagrams such as:

Production Example of HTGG

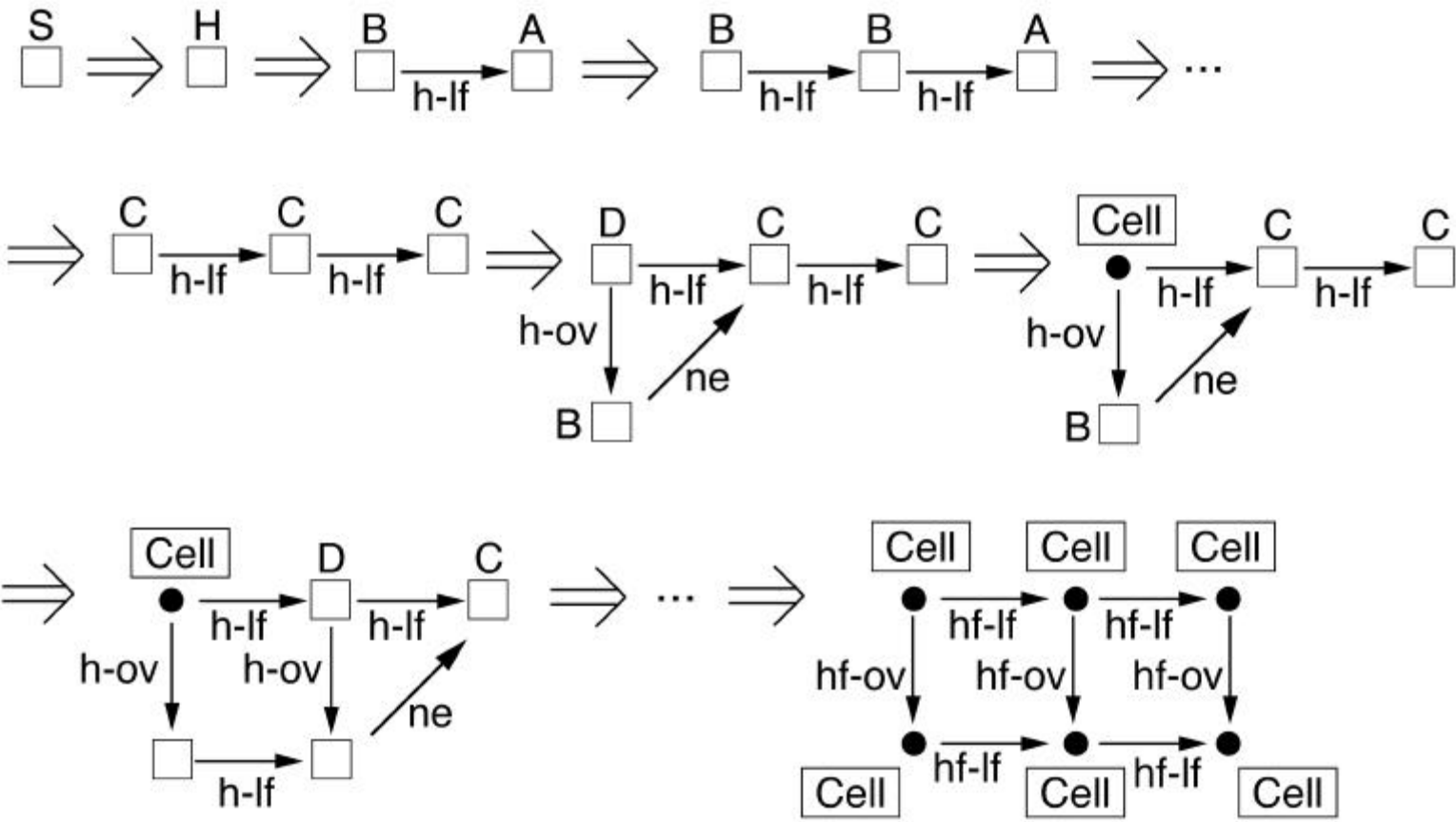


Features of HTGG

GG	Type	Rewriting Rule	Attribute Rule
HTGG	Context-sensitive	69	308



■ Derivation of HTGG





5. Diagram Processing System

KEYAKI – CASE2000 Concept

1. HichartED

Hichart program diagram editing component

2. HiTS

Hichart program diagram filtering component

3. LIVE

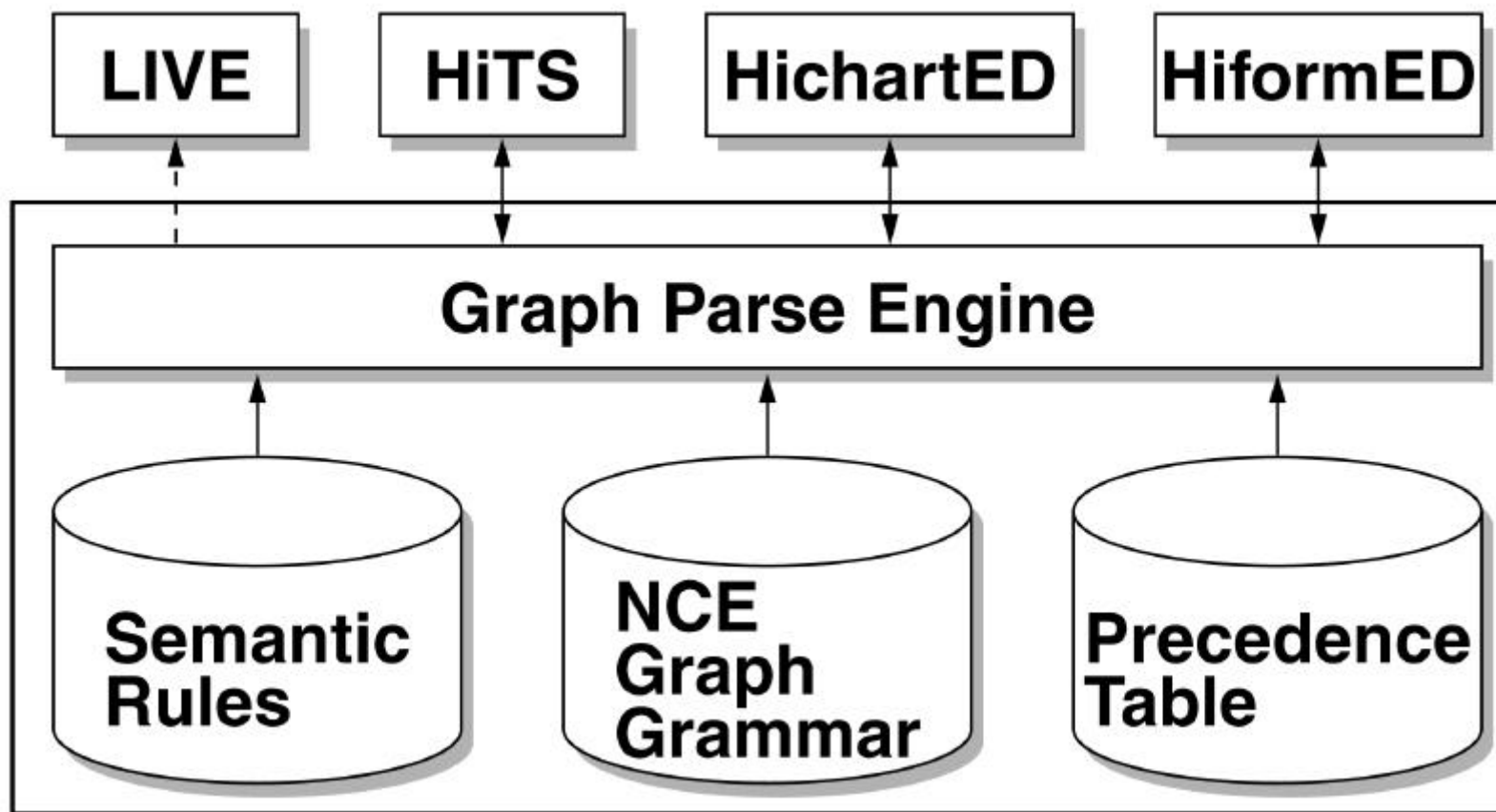
Program variable analyzing component

4. HiformED

Hiform diagram component

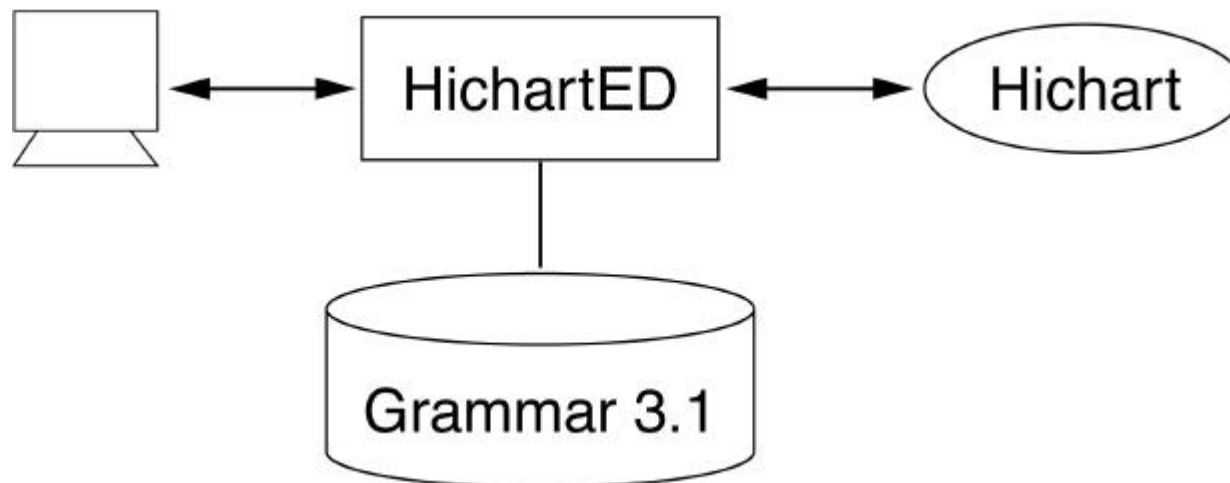


KEYAKI-CASE2000 Inside

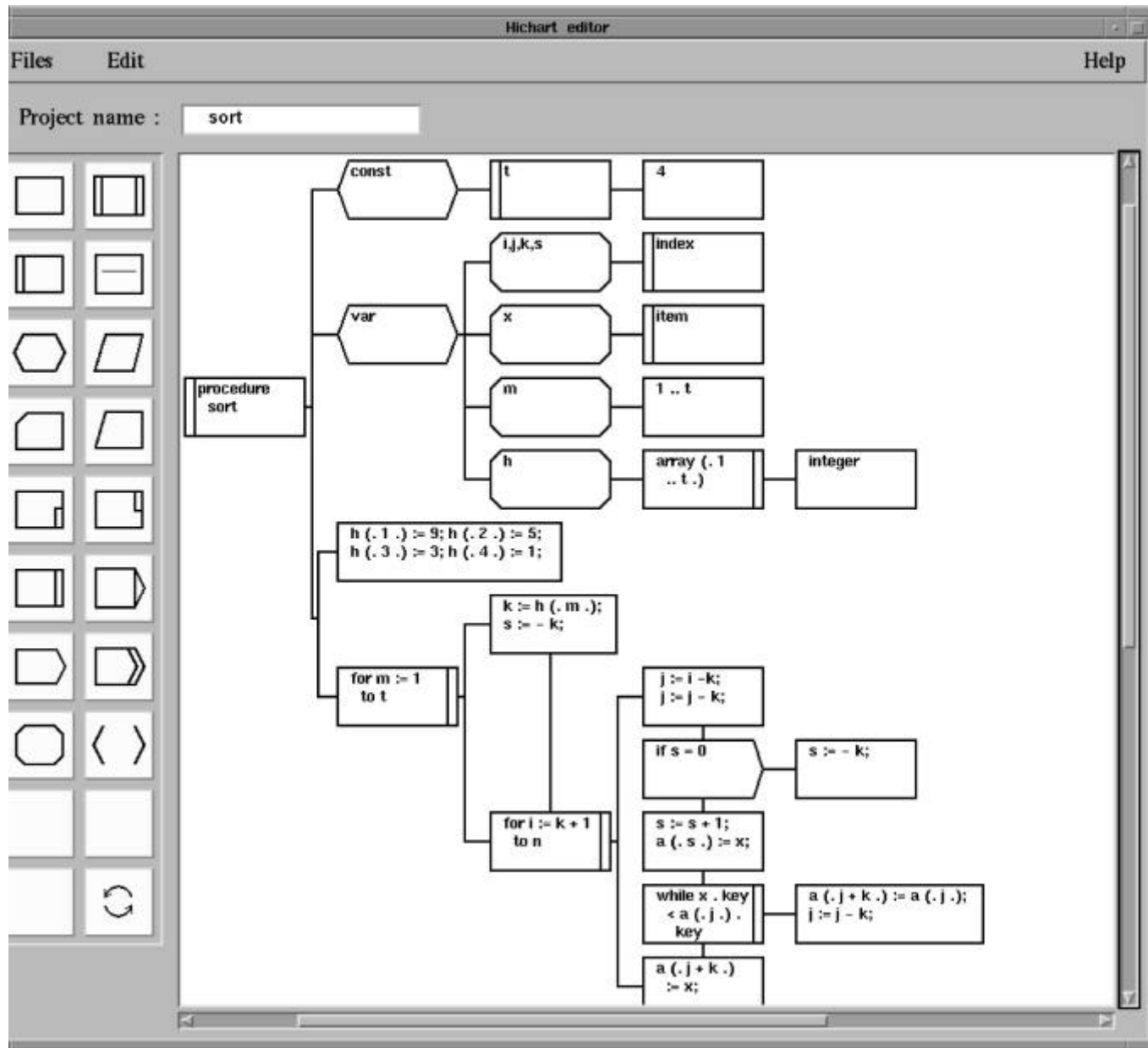


5.1 HichartED

- The Hichart program diagram editing component
- Syntax-directed diagram editor
- Editor-Commands defined by productions



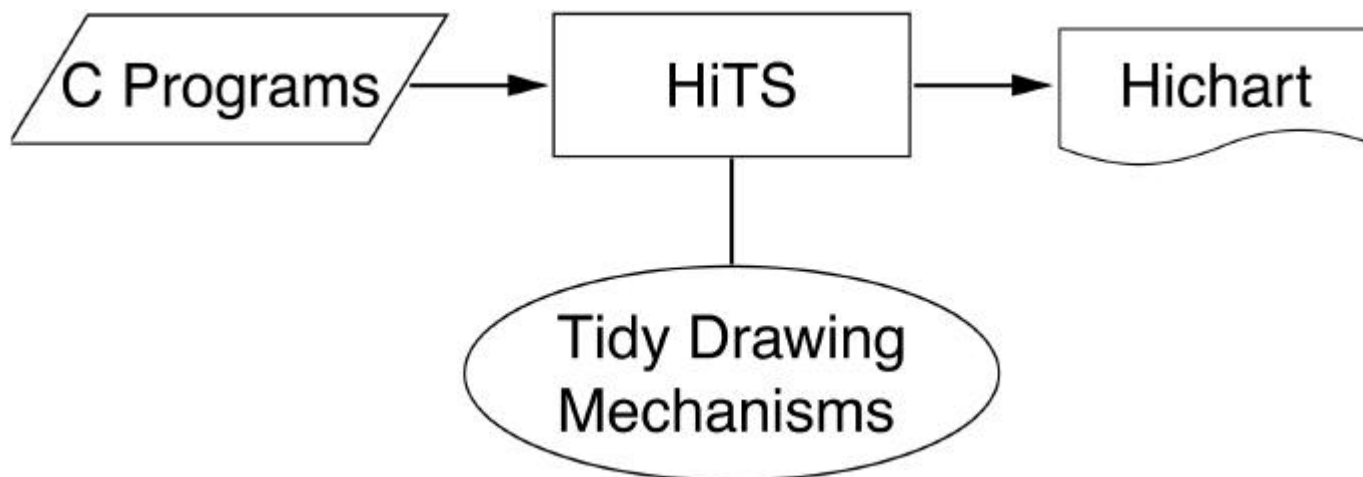
Screen Concept of HichartED



5.2 HiTS

Miyadera et.al [7,9]

- Program flowchart processing system
- Generating Hichart
- Based on a tidy drawing of trees
- Visualizing data structure and control flow



Execution Screen in HiTS

The screenshot displays the HiChart Translation Service interface within a web browser window. The browser's address bar shows the local file path: `files:/tmp_net/home/4j/a92e/`. The service page includes a menu (File, Edit, View, Go, Bookmarks, Options, Directory), a location field, and navigation links (What's New, What's Cool, Handbook, Net Search). The main content area is titled "HiChart Translation Service" and features a language selector (English / Japanese), an input field for a target source file (containing `~/dijkstra.g`), a "Translate" button, a layout type selector (set to "Layout ES"), and an input field for cell size (horizontal length: 300, vertical length: 300). A "Select objects if you need" section lists various chart types, with "HiChart flowchart (go file)" selected. A "Reset" button is located at the bottom of the form.

Below the browser window, the execution screen displays a complex flowchart and a code editor. The flowchart illustrates the execution of a Dijkstra's algorithm, showing the flow of control and data between various components. The code editor shows the following C code:

```
#include <stdio.h>
#include <limits.h>

typedef enum {FALSE, TRUE}
bool;

#define INF INT_MAX
#define N 8
#define START 0

int In[N], Id[N];

void dijkstra(int start,
{
    bool qlist[N];
    int i, current, n;

    for (i = 0; i < N; i++)
        qlist[i] = FALSE;
        Id[i] = INF;
        In[i] = -1;
}
qlist[start] = TRUE;
account = 1;
Id[start] = 0;

do {
    min = INF;
    for (i = 1; i < N; i++)
        if (Id[i] < min) {
            min = Id[i];
            current = i;
        }
    for (i = 1; i < N; i++)
        if (qlist[i] == FALSE && Id[i] < INF)
            Id[i] = min + cost[current][i];
} while (current != -1);

for (i = 1; i < N; i++)
    printf("%d ", Id[i]);
printf("\n");
}
```

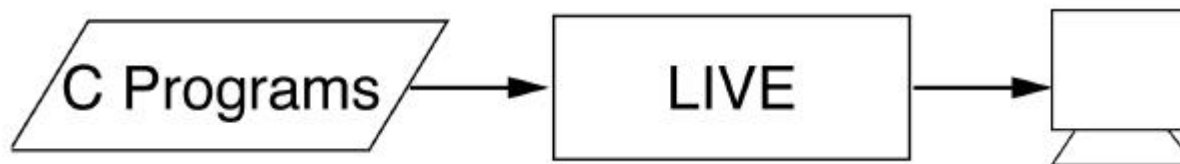
The execution screen also features a large, stylized pair of eyes in the upper right corner, and a status bar at the bottom right indicating "Auto save file".

Computer Software Lab (Data Base Division)
Yousuke Miyashiro < e-mail: miyashiro@db.dendai.ac.jp >

5.3 LIVE

Miyadera et.al. [9]

- A program variable analyzing component
- Developed to support the program modularization
- The user can modify the data structure and modules



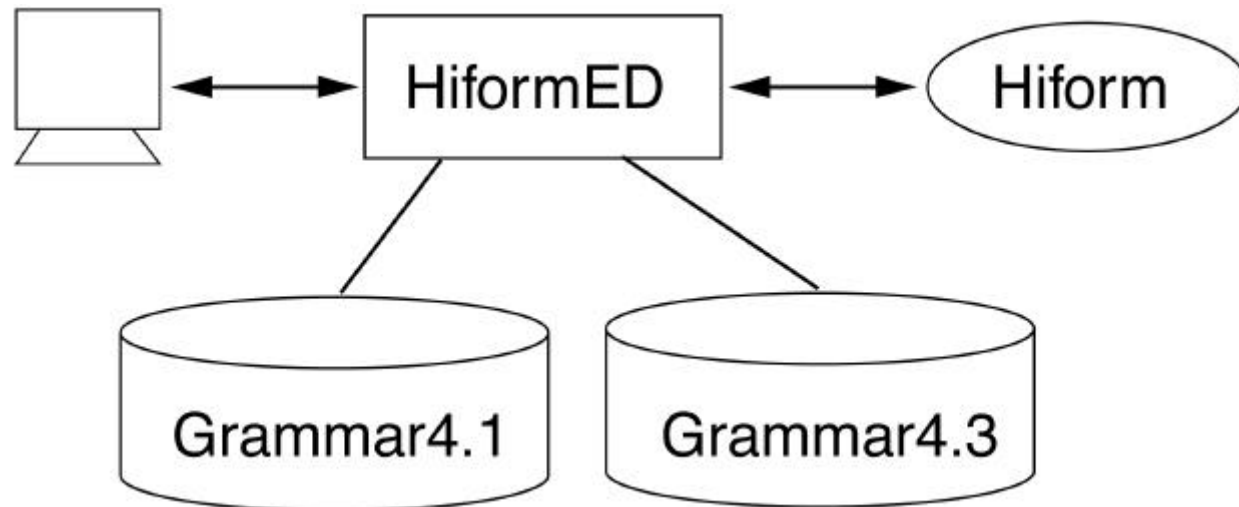
Execution Screen in LIVE

The screenshot displays the LIVE (Language for Visualizing and Interpreting) execution environment. It features several windows and components:

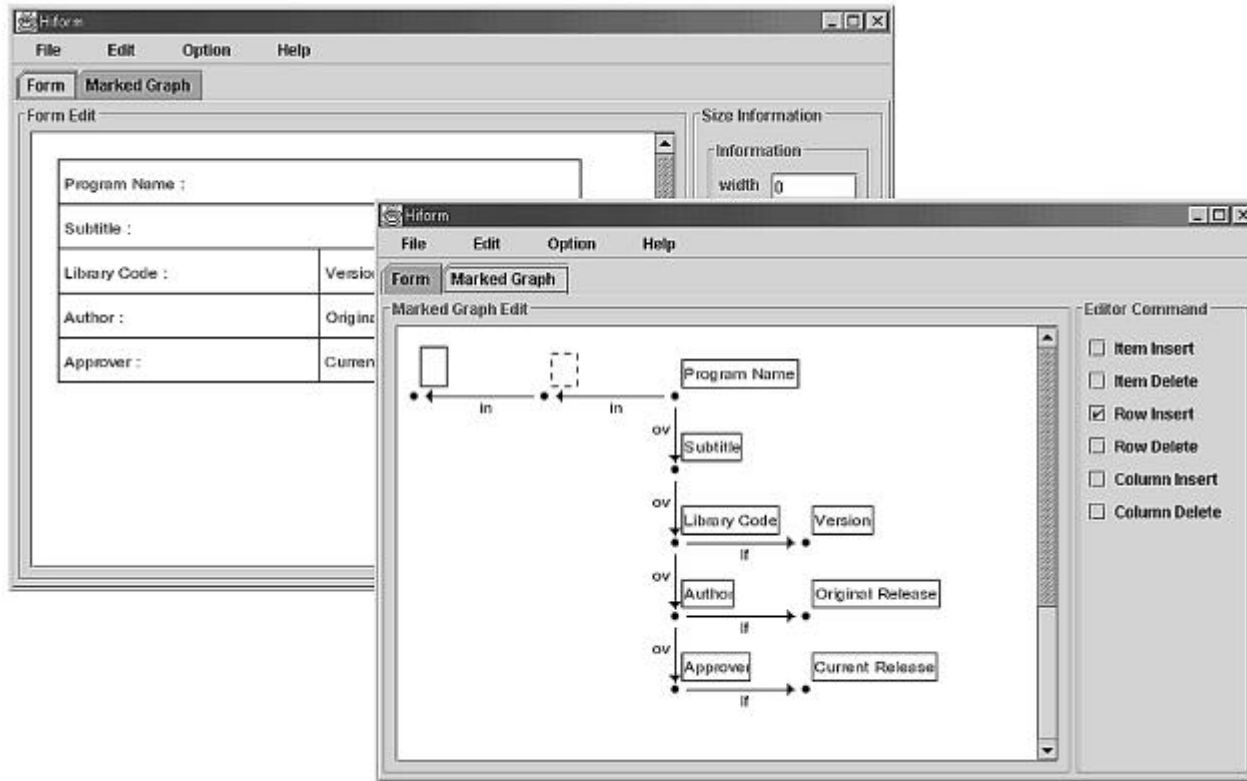
- Top Left:** A gauge and a waveform plot labeled "issun01".
- Top Center:** A large cartoon face icon.
- Top Right:** A file browser window titled "ファイル" (File) showing a directory structure with files like "matriboul.liv", "matriboul.prt", "paper17", "screen1-1.ps", and "screen1.ps".
- Left Panel:** A "File" window containing the source code for a program named "MatribMul". The code includes declarations for arrays A, B, and C, and a nested loop structure for matrix multiplication.
- Bottom Left:** A "File" window showing the execution progress, with line numbers from 00010 to 00048 and corresponding execution status (e.g., "I", "I J", "I J Element").
- Right Panel:** A data flow graph (DFG) titled "W, version 1.5". It visualizes the execution of the code, showing nodes for array elements and operations, connected by arrows representing data flow.


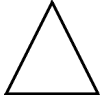
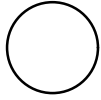
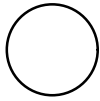
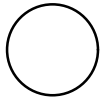
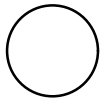


5.4 HiformED

- Hiform Diagram Processing Component
- A Java application



Screen Concept of HiformED



System	NCE GG	System with NCE GG	System without NCE GG
HichartED			
HiTS		Not yet	
LIVE	Not yet	Not yet	
HiformED		Not yet	



6. Conclusion

Summary

- We proposed an attribute NCE grammar as a universal model of visual processing of diagrams.
- Processing methods were also considered.

Future Work

- We are developing diagram processing system now.



6. Conclusion

System	NCE GG	System with NCE GG	System without NCE GG
HichartED	△	△	○
HiTS	○	Not yet	○
LIVE	Not yet	Not yet	○
HiformED	△	Not yet	△

Our Project Web Site :

Including detailed description of Graph Grammars

■ **URL:**

<http://www.hichart.org/>