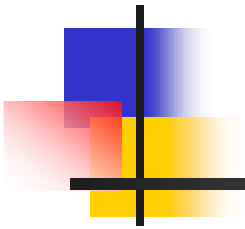


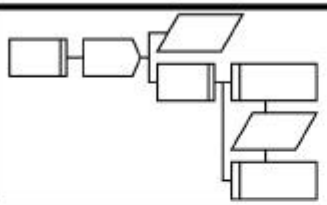
Syntactic Processing of Diagrams by Graph Grammars

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

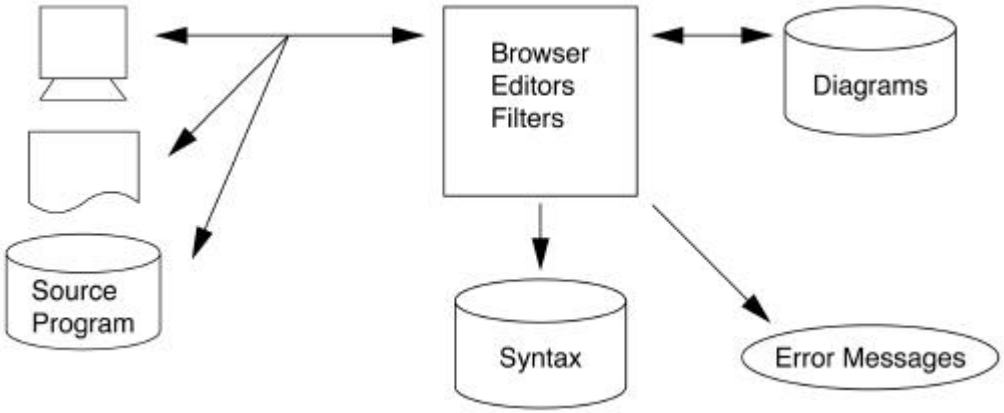
Youzou MIYADERA, (Tokyo Gakugei Univ.)
Kimio SUGITA, (Tokai Univ.)
Kensei TSUCHIDA (Toyo Univ.)

Abstract

- **TARGET:** Diagrams in Software Specification

Diagrams		Corresponding Graphs	Universal Models
Hierarchical Diagram		Attribute Tree	Attribute NCE CFGG
Nested Diagram	Program Code: / Program Name: / Library Code: / **Program Specification** / Version:	Attribute Marked Tree	Attribute NCE CFGG
Tessellation Diagram	Name: x, y; Type: int, float; Size: 2, 4; G/L: G, L	Attribute Marked Tessellation Graph	Attribute NCE CSGG

- **GOAL:** Syntactic Processing



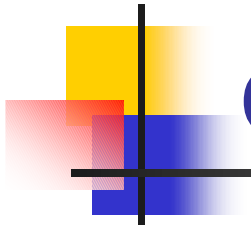
Abstract (continued)

- **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

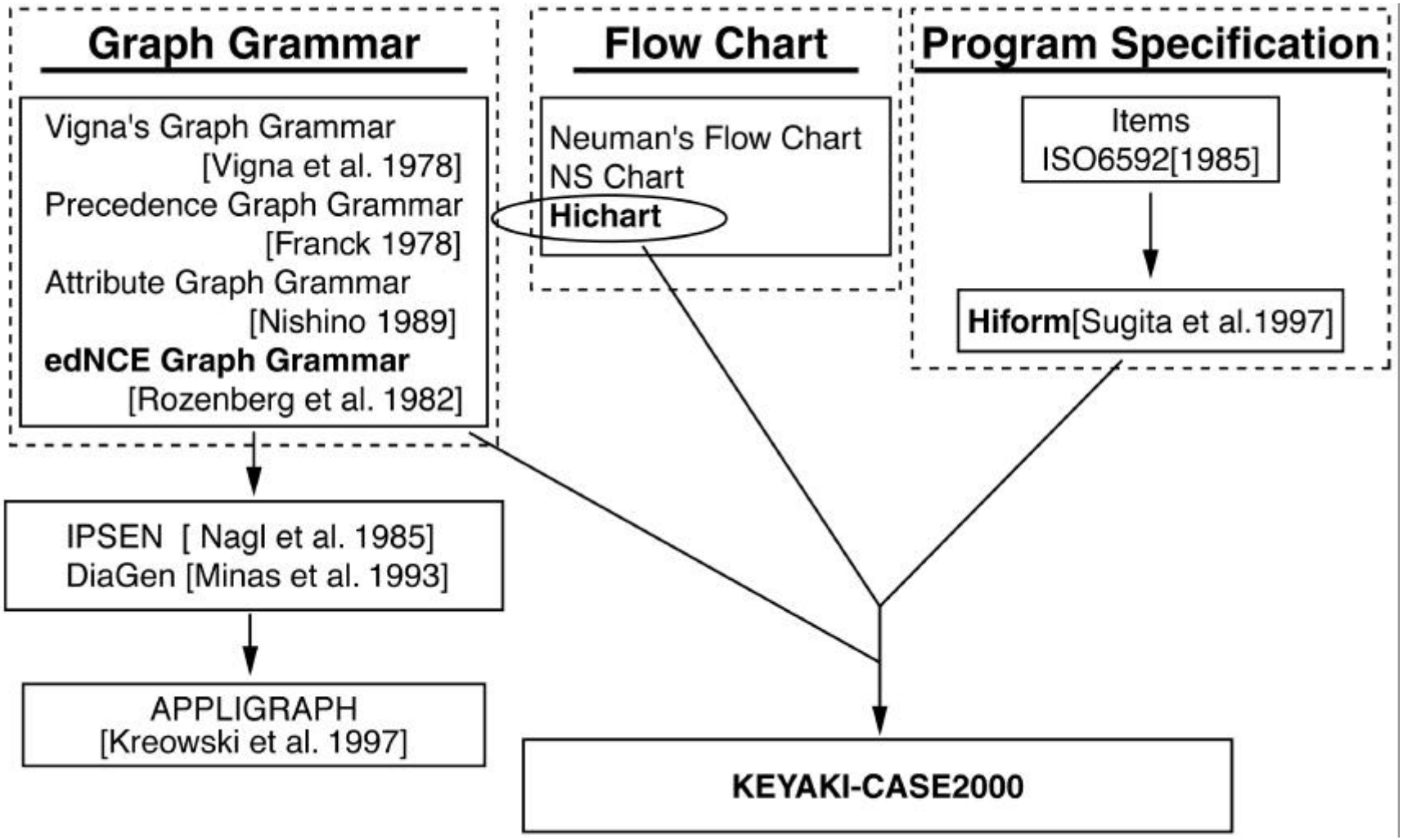
  - **Syntactic Processing Methods of Diagrams**

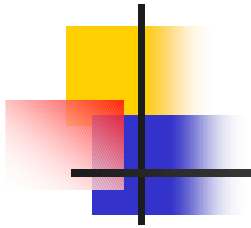


Contents

1 Introduction

Background





Our History

1978 Graphical notation of program flowchart Hichart

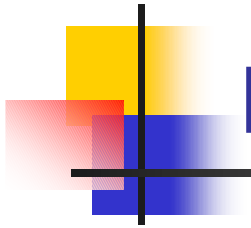
1987 Hichart (non-syntactic)[COMPSAC 11]

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]

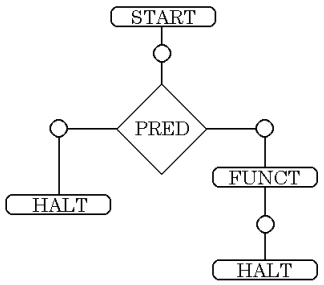
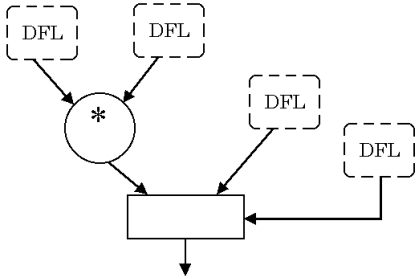
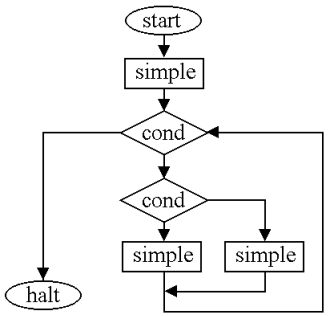
2000 Syntactic Processing of Diagrams by Graph Grammars[IFIP WCC 2000]

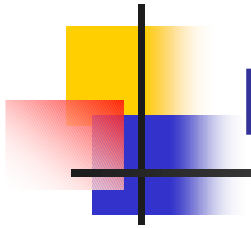


Related Works

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

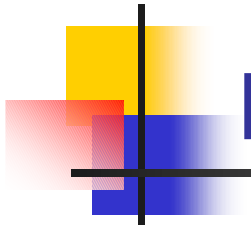
Related Works (continued)

Diagrams	Known Models	
Flowchart 	CF PLEX Grammar	K.S.Fu (1982)
	Relational Grammar	K.Wittenburg Et al.(1991)
Data Flow Chart 	Positional Grammar	G. Costagliola et al. (1990)
Structured Flow Chart 	Symbol Relation Grammar	F.Ferruci et al. (1996)



Motivation

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



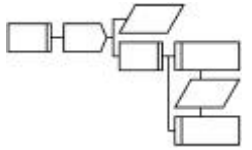
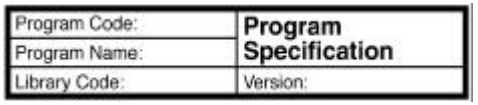
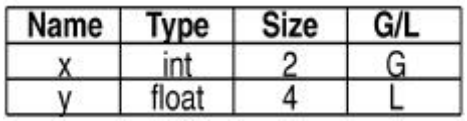
Purpose

- To *characterize* types of graph grammars that generate three types of diagrams

- To propose *integrated processing methods* of diagrams using the graph grammars

Results

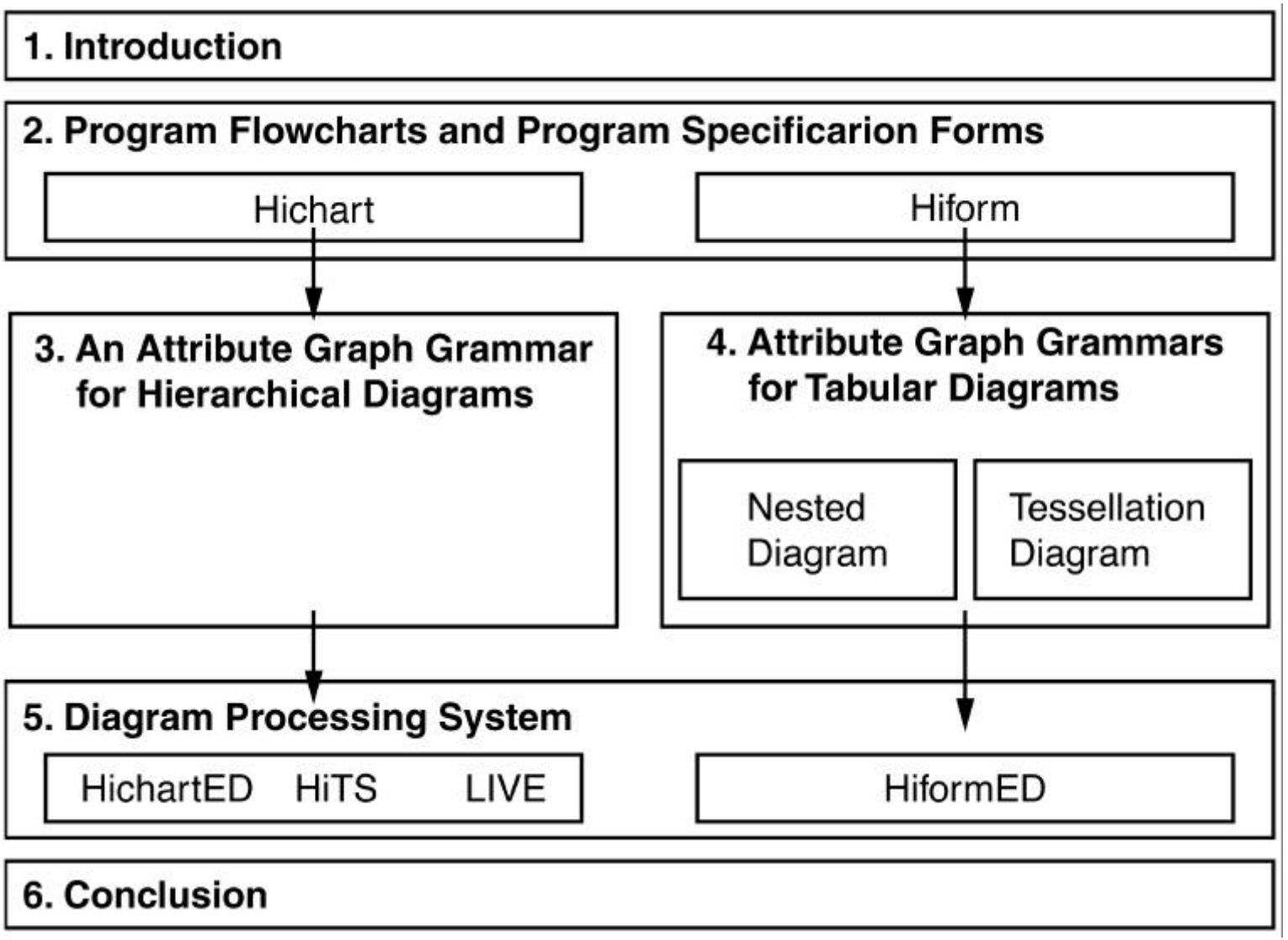
- Types of graph grammars

Graph Grammar	Grammar's type (Rewriting rule, Attribute rule)	Diagram
HCGG	*Context-free* (67, 723)	
HNGG	*Context-free, precedence* (280, 1248)	
HTGG	*Context-sensitive* (69, 308)	

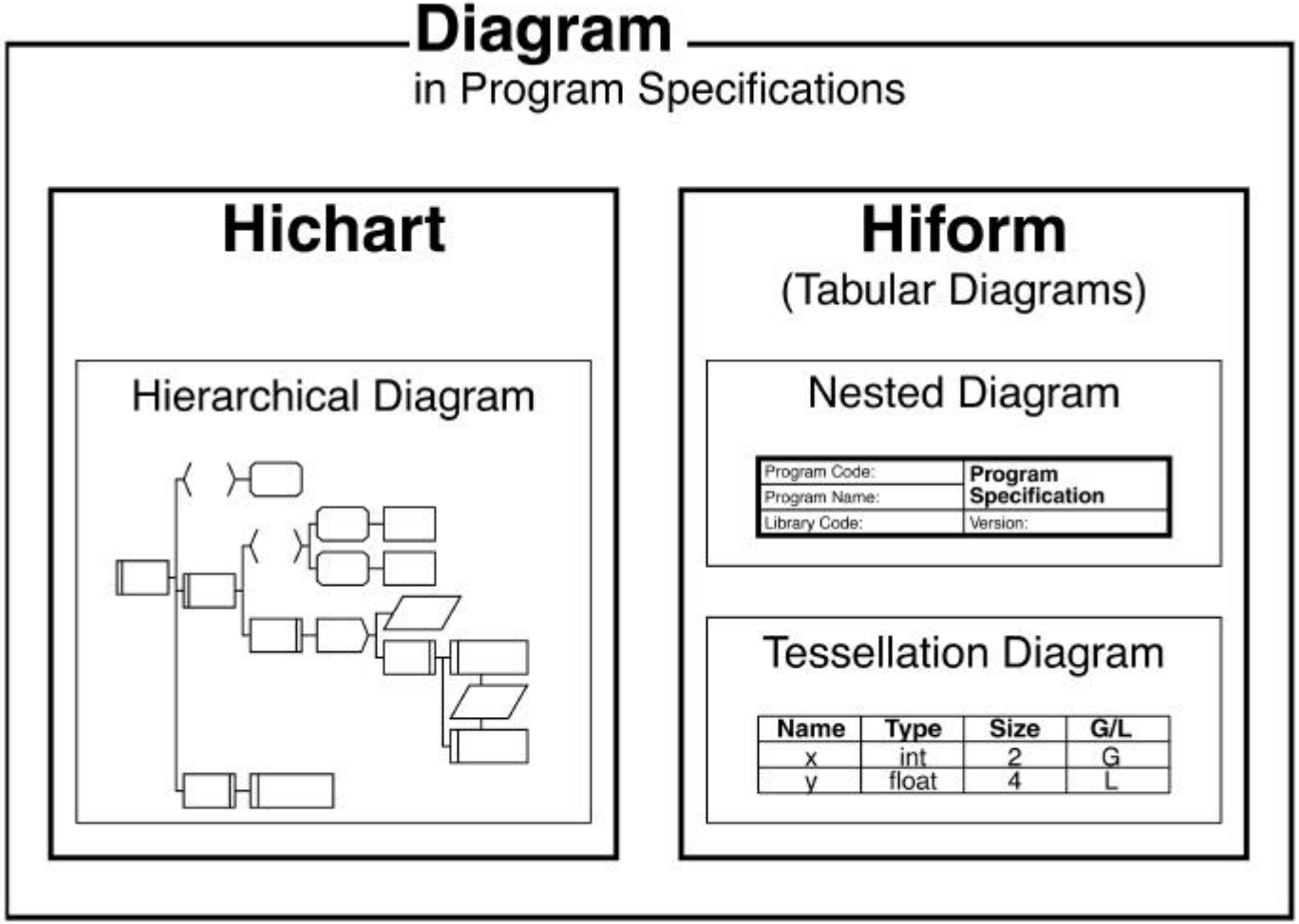
- Integrated processing methods of diagrams

  Diagram Processing System **KEYAKI-CASE2000**

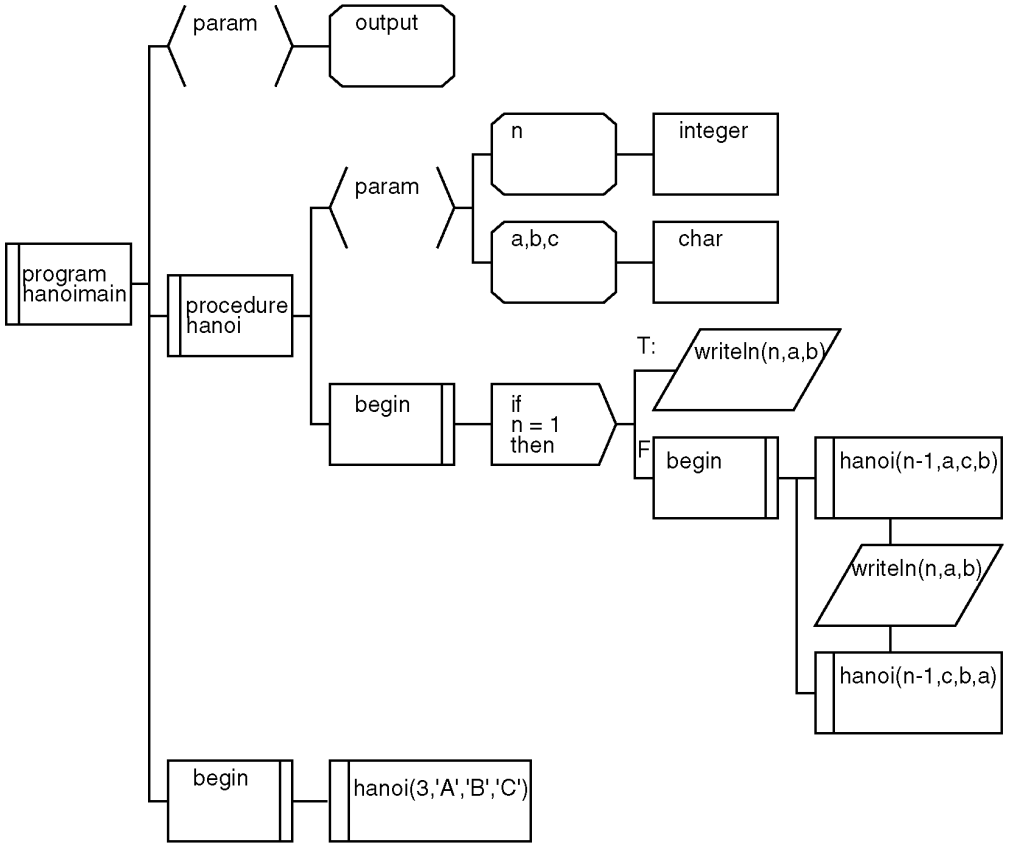
Contents


1. Introduction
2. Program Flowcharts and Program Specificarion Forms
Hichart
Hiform
3. An Attribute Graph Grammar for Hierarchical Diagrams
4. Attribute Graph Grammars for Tabular Diagrams
Nested Diagram
Tessellation Diagram
5. Diagram Processing System
HichartED HiTS LIVE
HiformED
6. Conclusion

2. Program Flowcharts and Program Specification Forms



2.1 Hierarchical Diagram for Program Flowchart



A Hichart program flowchart (Tower of Hanoi).

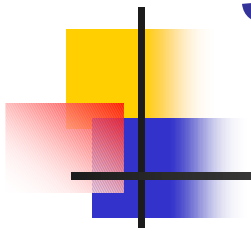
2.2 Tabular Diagrams for Program Specification Forms

Hiform

(a program specification language)

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

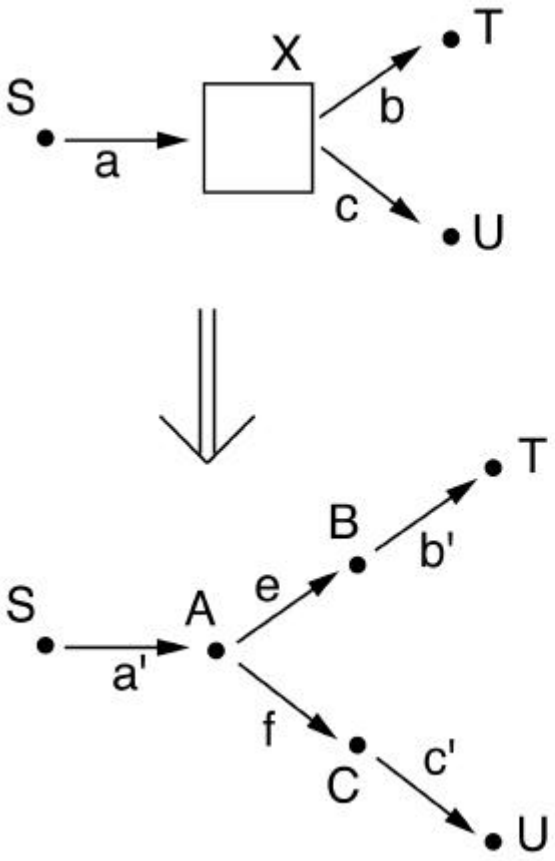
Project Code:	A 5
Program Name:	**Program Specification-1** p
Library Code:	Version:
Author:	Original Release:
Approver:	Current Release:
Problem Description:	
Problem Supplementary Information (Theoretical Principles, Methods and References):	
Problem Solution: 1.Conventions and Terminology 2.Principles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



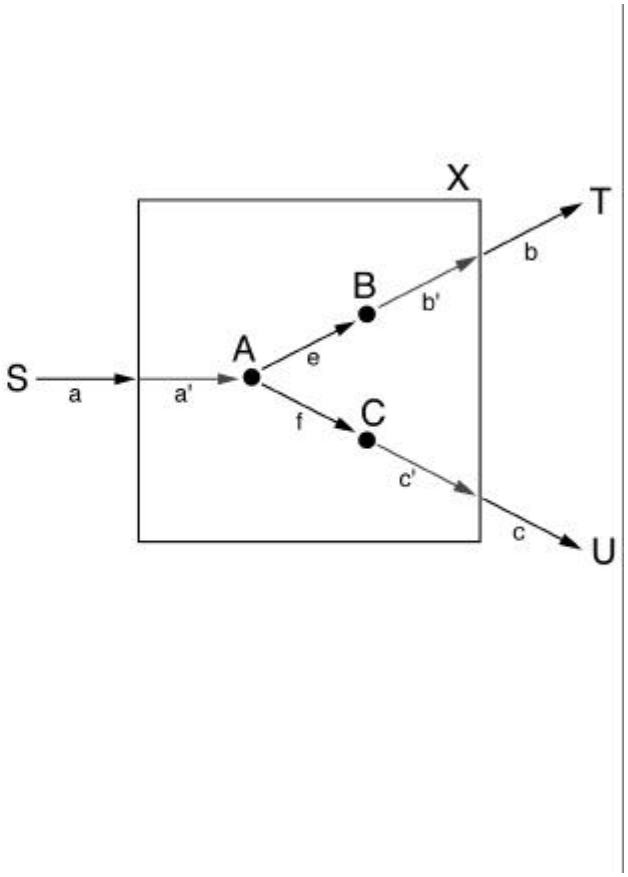
3. An Attribute Graph Grammar for Hierarchical Diagrams

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

Derivation :

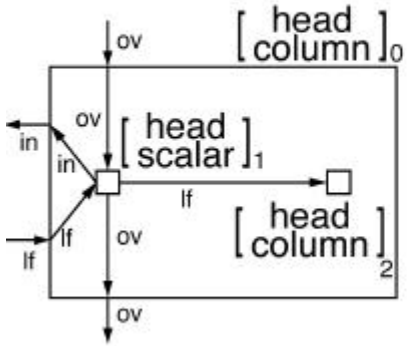


Rewriting Rule:



Attribute Context-free NCE Graph Grammar (continued)

Production with attribute rules:

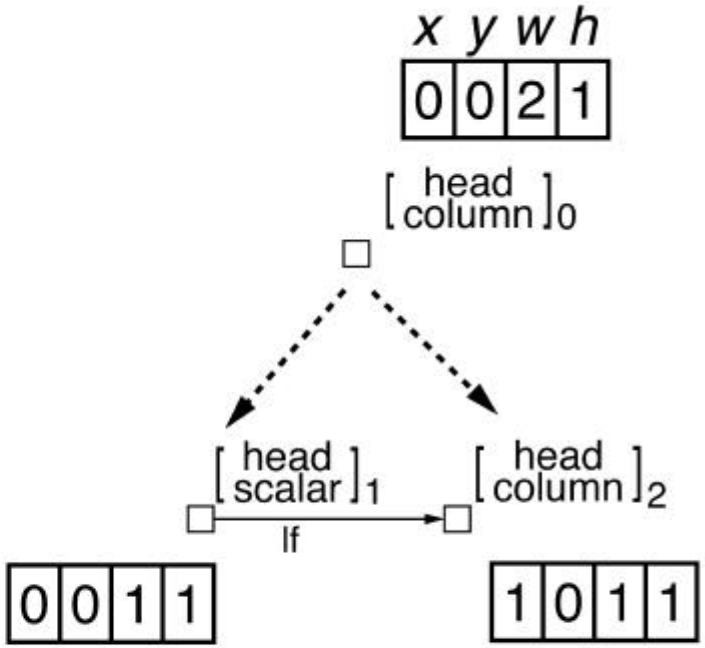


Semantic Rule

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 :

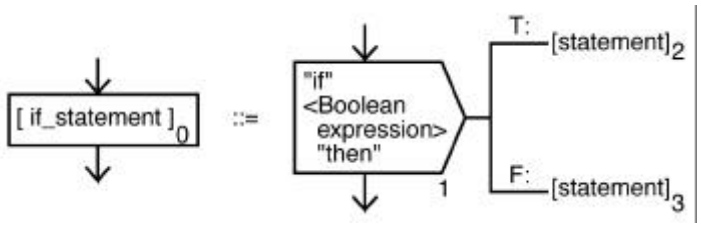


Remark: Our Attribute context-sensitive
NCE Graph Grammar rewrites exactly one
Node in each derivation [cf. Adachi-Yaku (1999)].

Grammar 3.1 HCGG

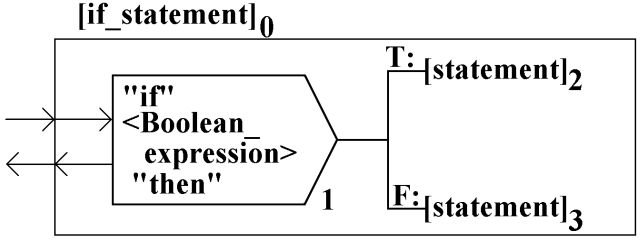
HCGG (HiChart Graph Grammar) is an attribute context-free NCE graph grammar for hierarchical diagrams in Hichart such as :

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])

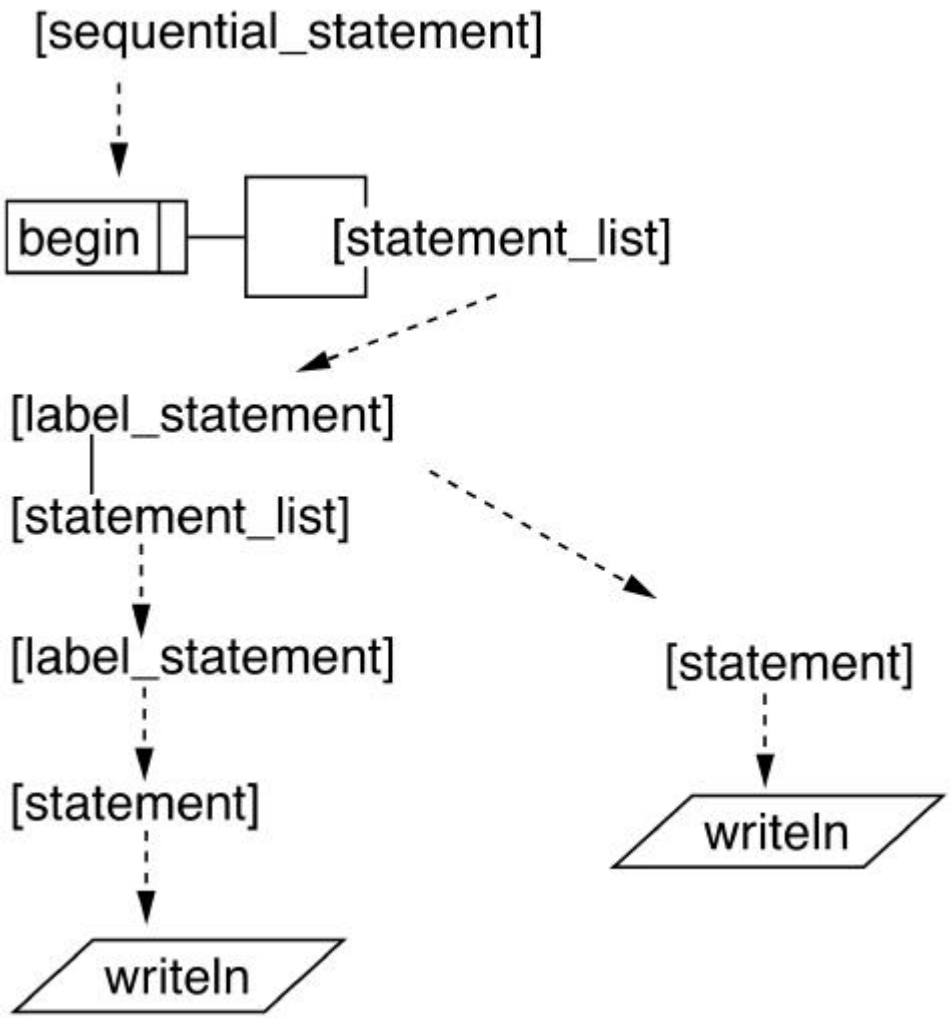
Features of HCGG

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

- **Property 3.2**

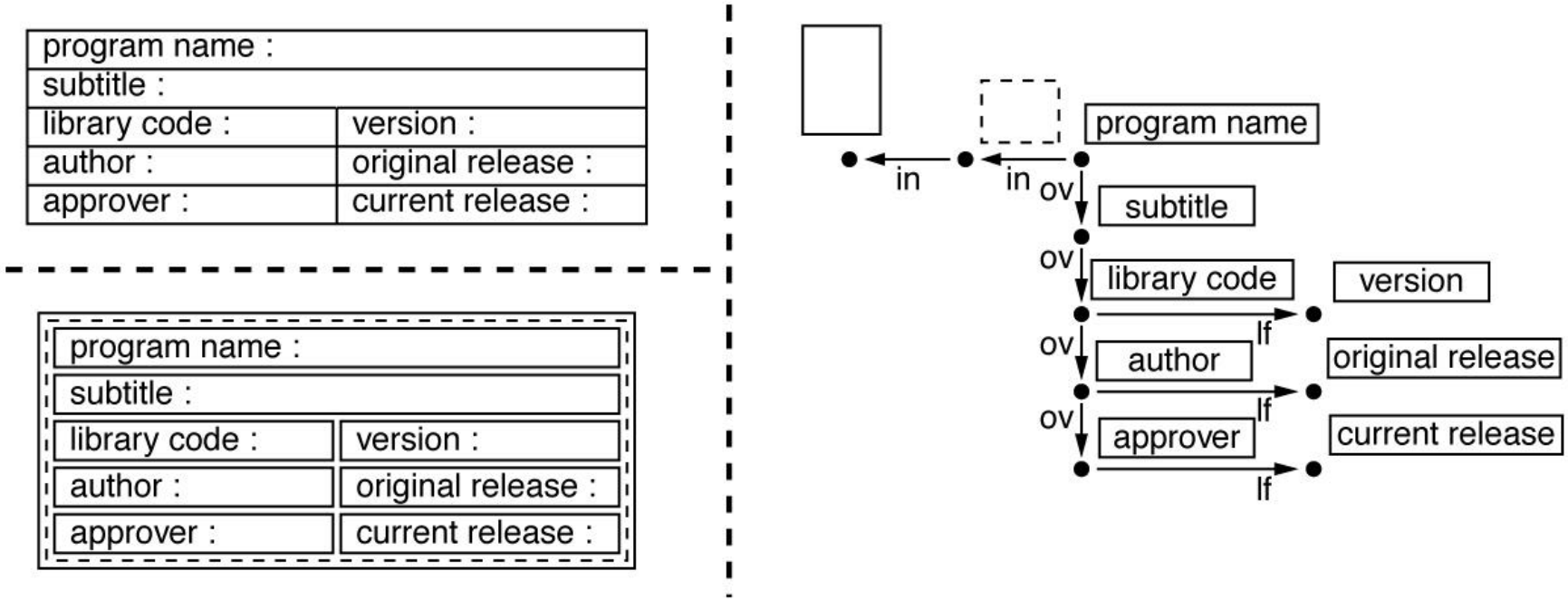
Attribute rules in HCGG are evaluated in linear time.

Derivation Tree of HCGG



4. Attribute Graph Grammars for Tabular Diagrams

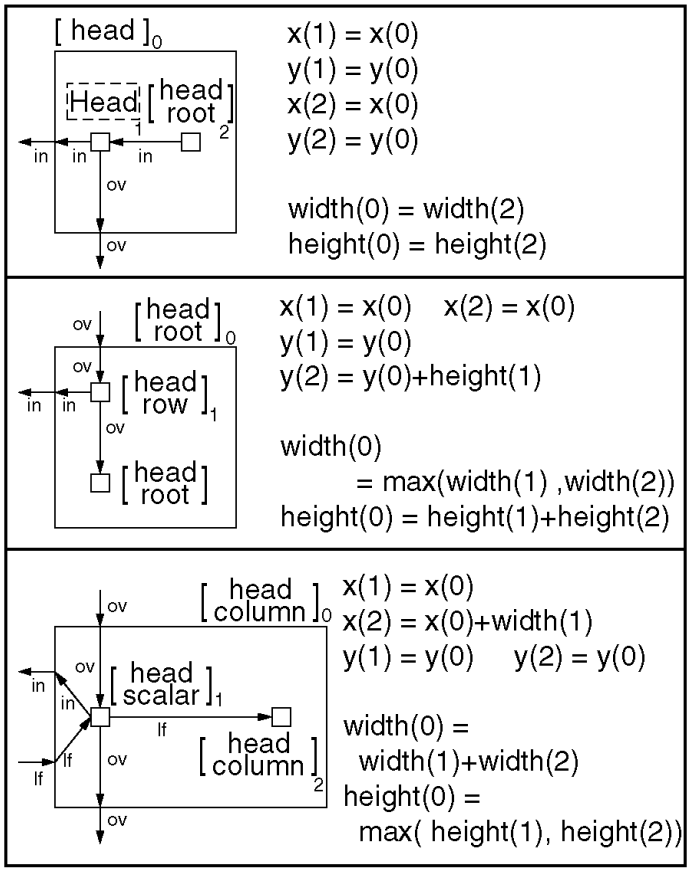
Nested Diagram and Its Corresponding Marked Graph

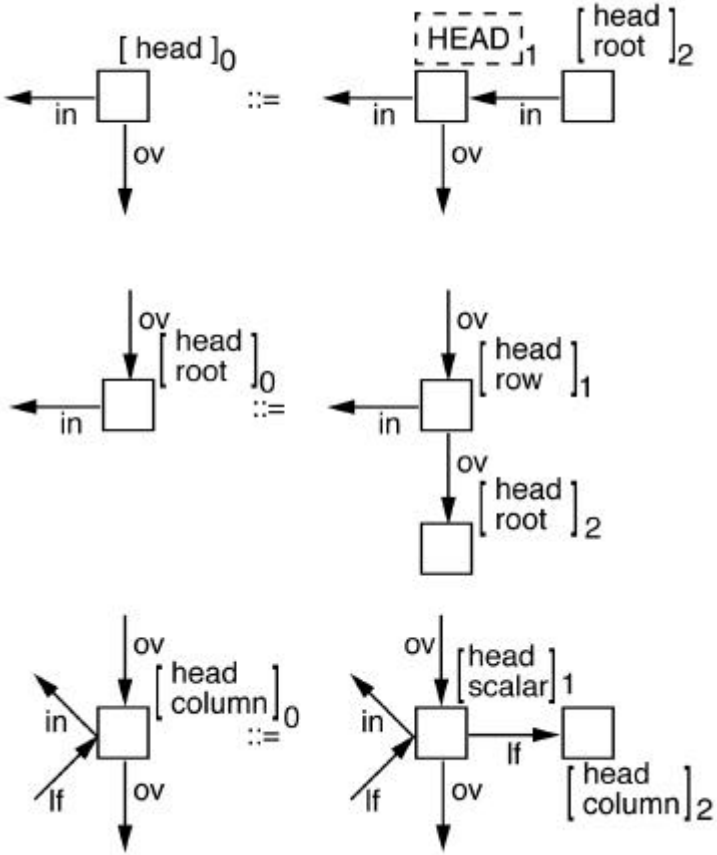


Grammar 4.1 HNGG

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

Production Examples of HNGG

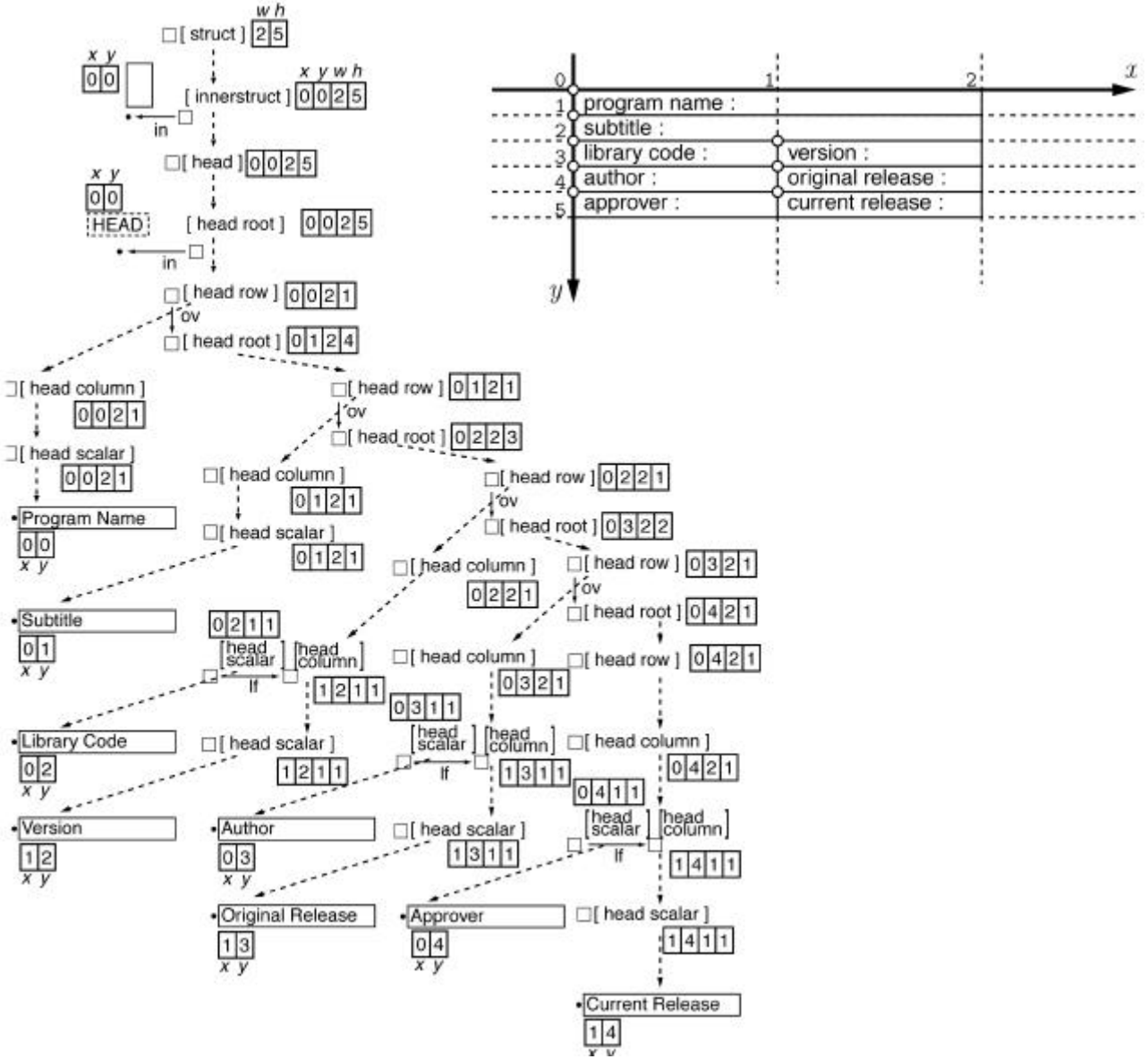




Features of HNGG

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

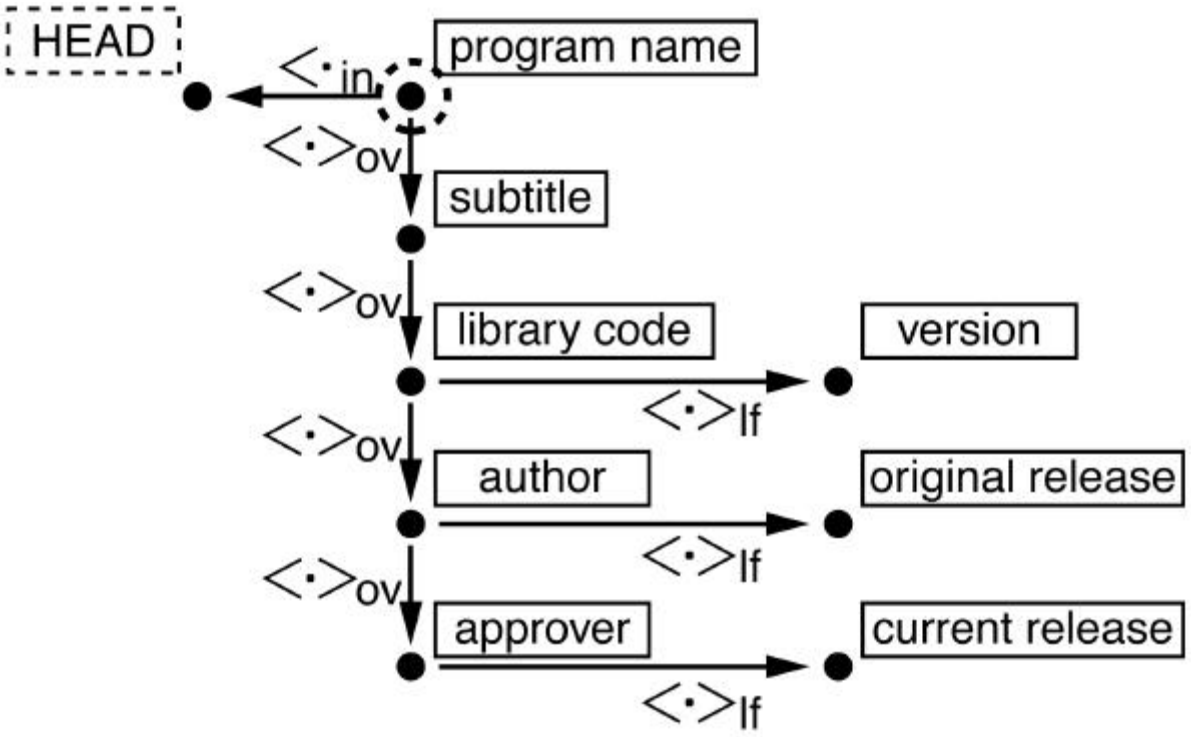
Derivation Tree of HNGG



- **Property 4.2**

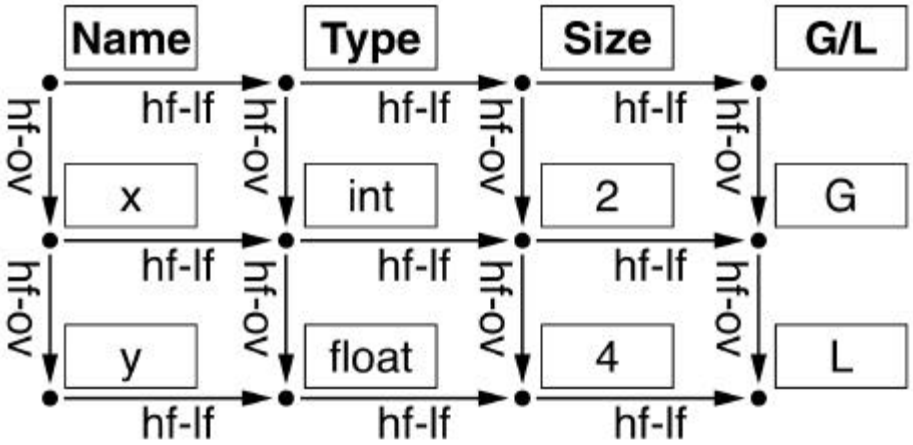
HNGG is a **precedence graph grammar** (see e.g. Franck 1978).

How to use precedence rule



Tessellation Diagram and Its Corresponding Graph

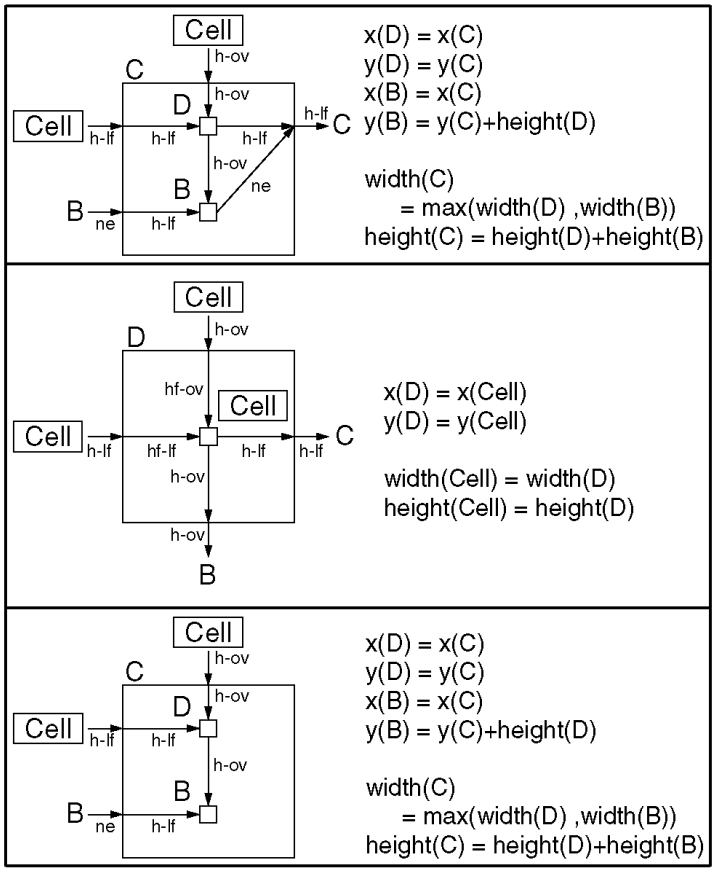
Name	Type	Size	G/L
x	int	2	G
y	float	4	L

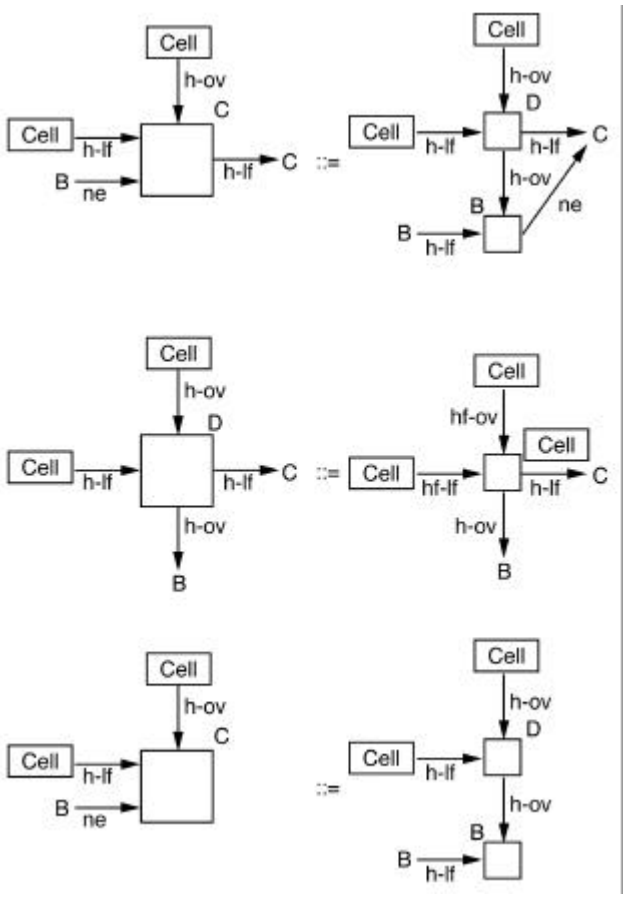


Grammar 4.3 HTGG

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

Production Example of HTGG

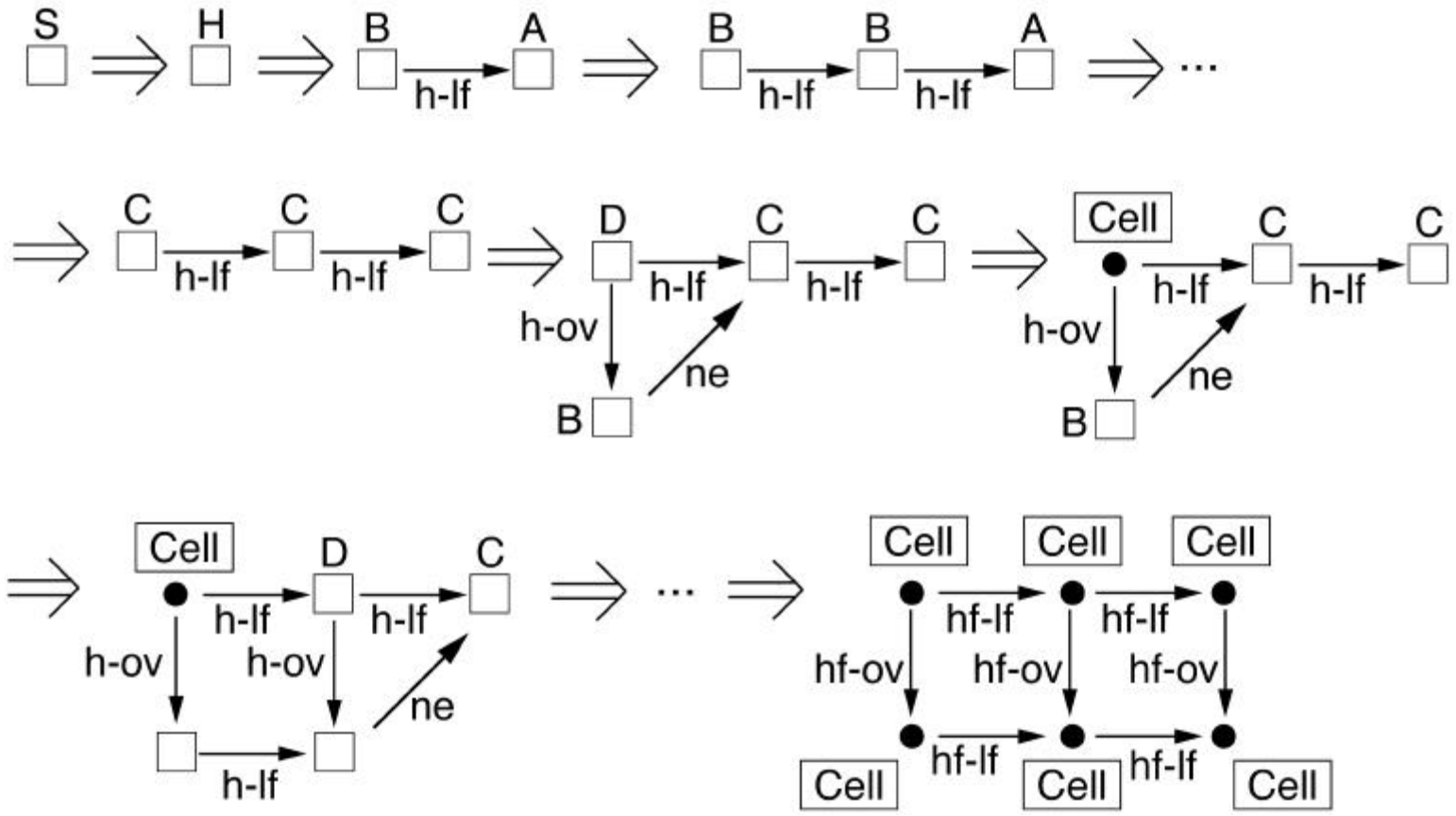


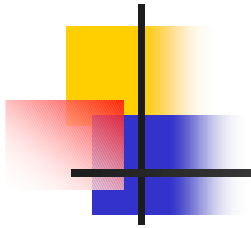


Features of HTGG

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

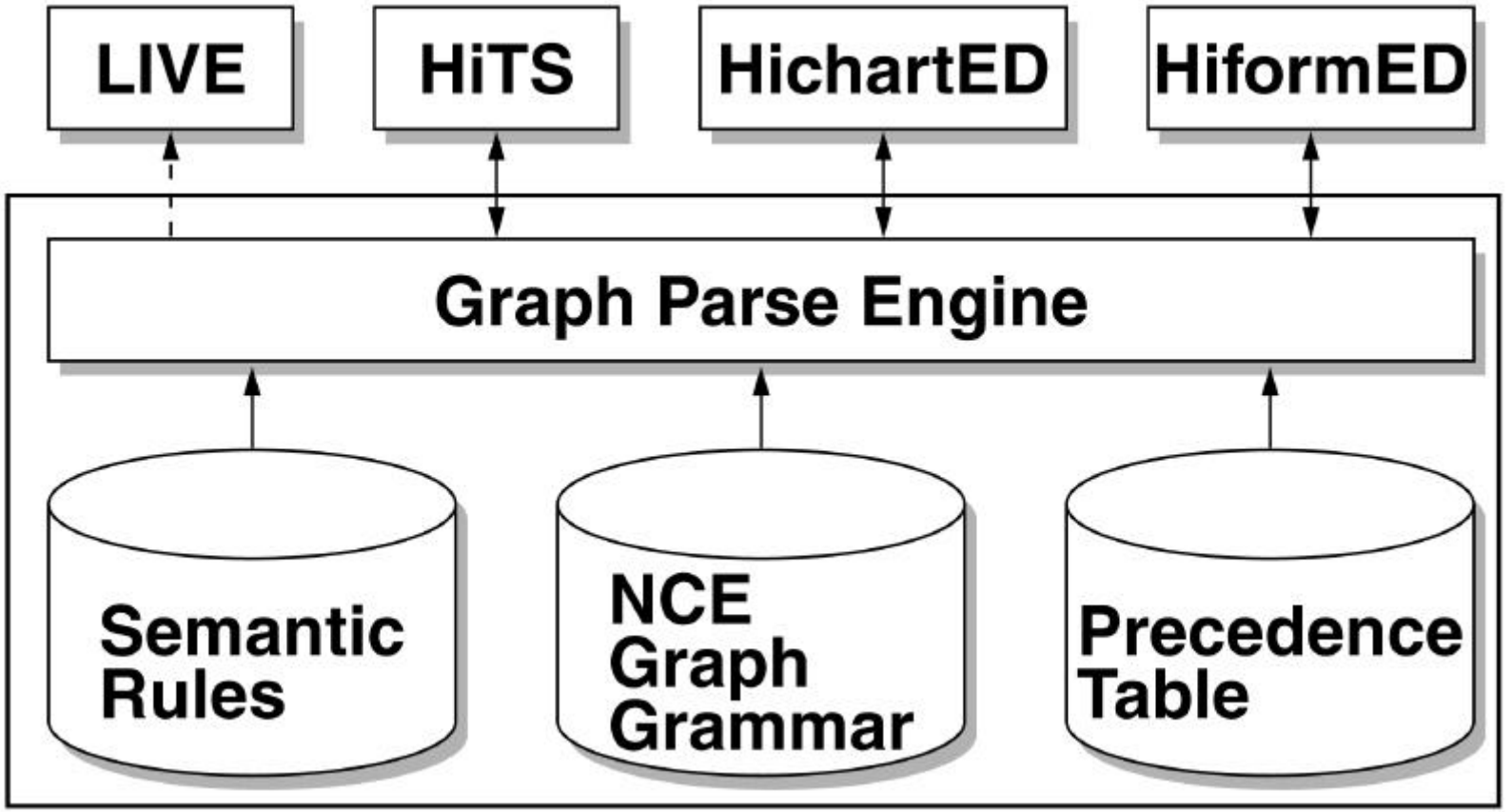
3. **LIVE**

   Program variable analyzing component

4. **HiformED**

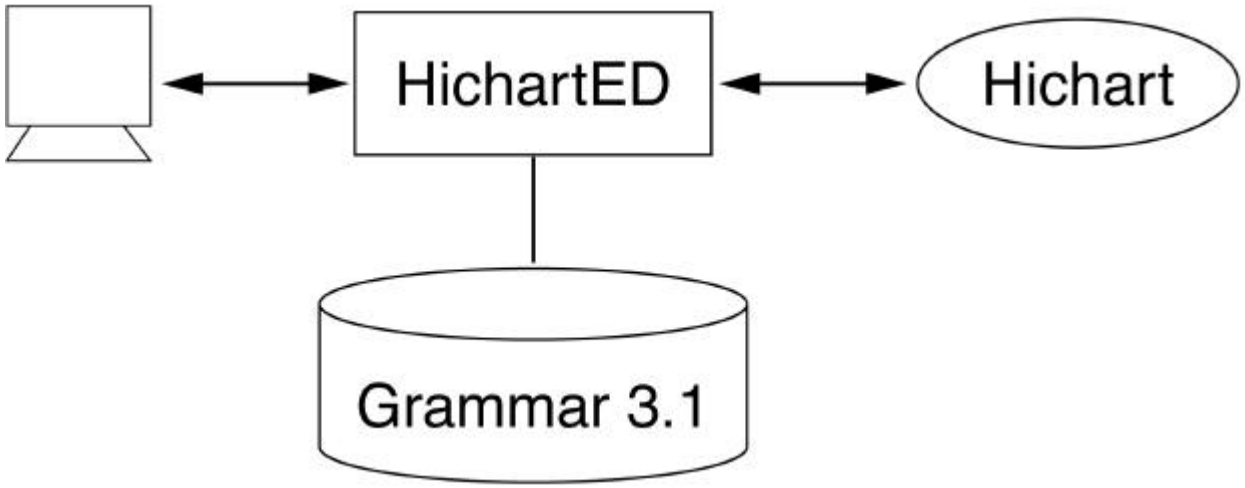
   Hiform diagram component

KEYAKI-CASE2000 Inside

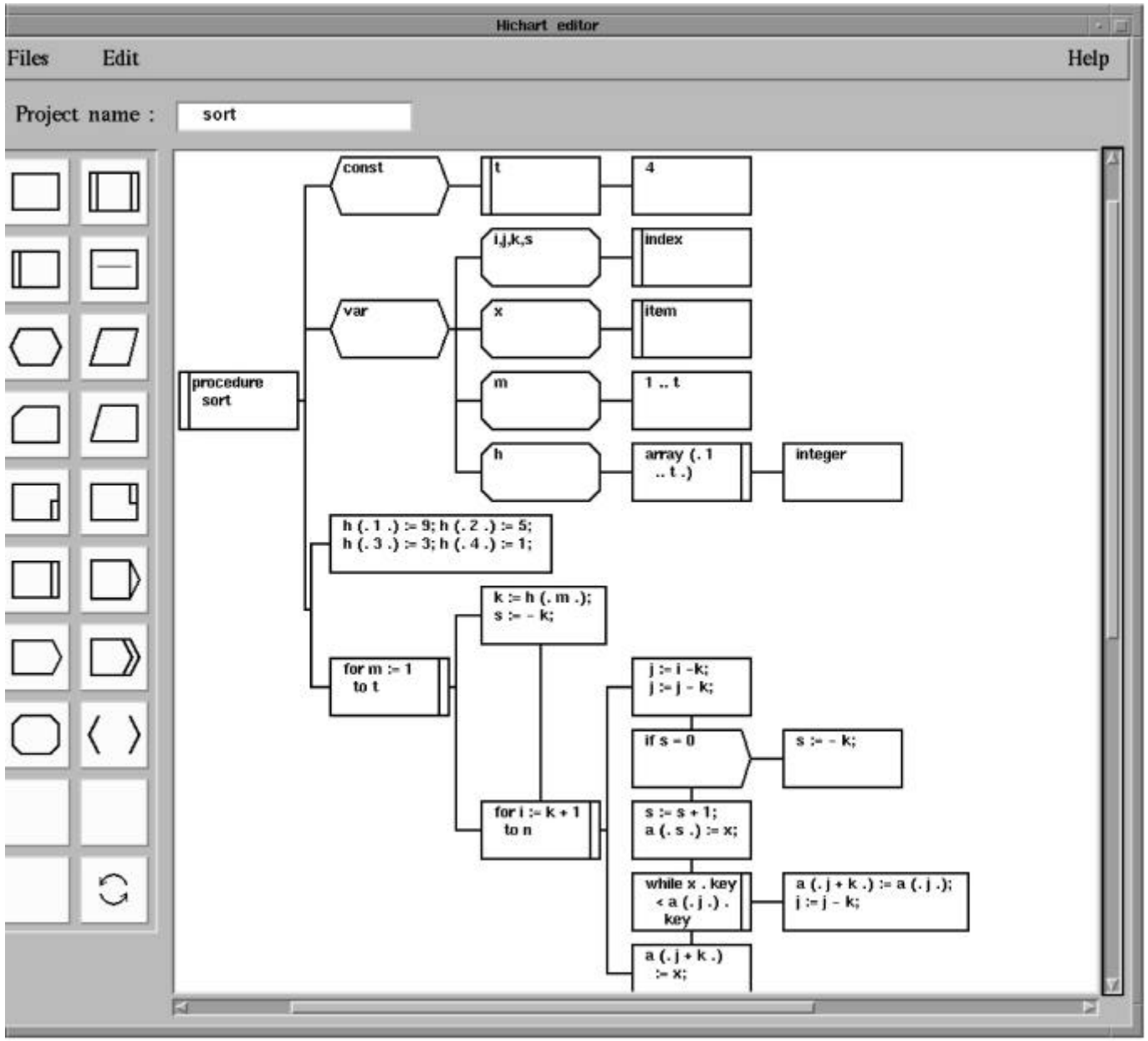


5.1 HichartED

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



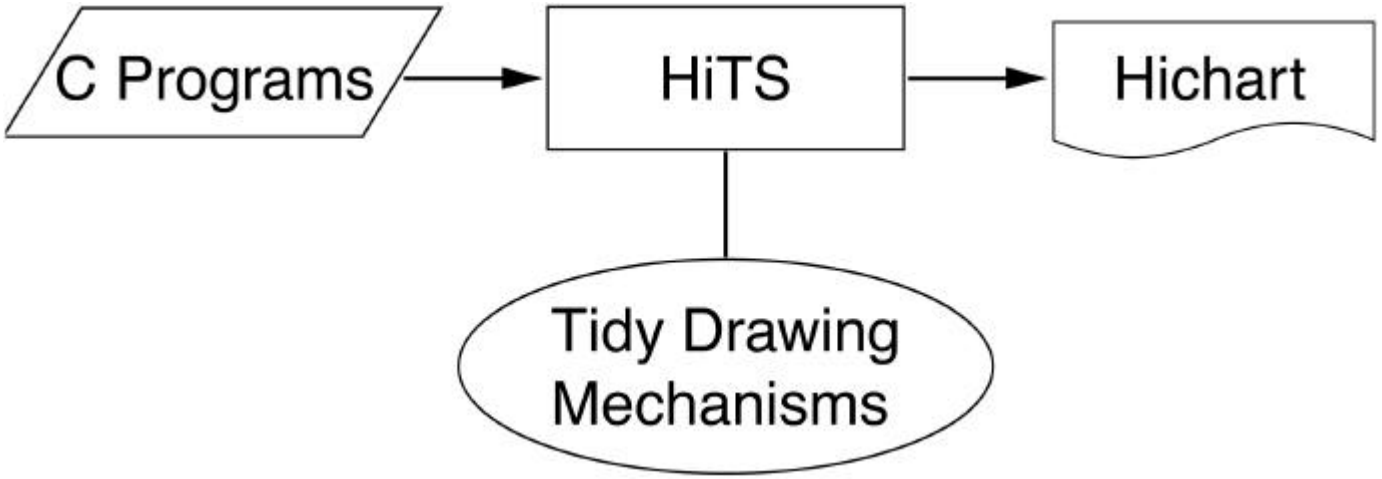
Screen Concept of HichartED



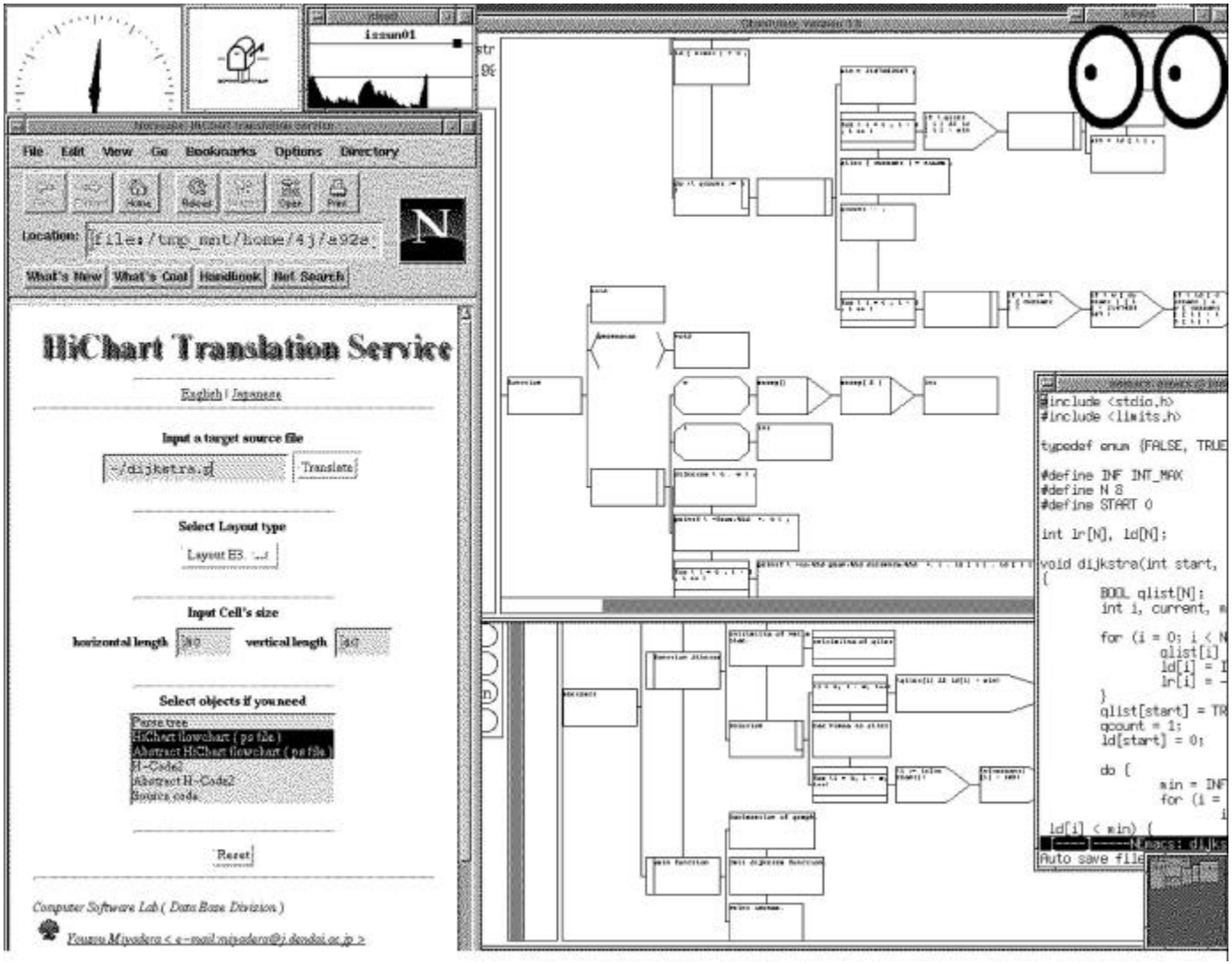
5.2 HiTS

Miyadera et.al [7,9]

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



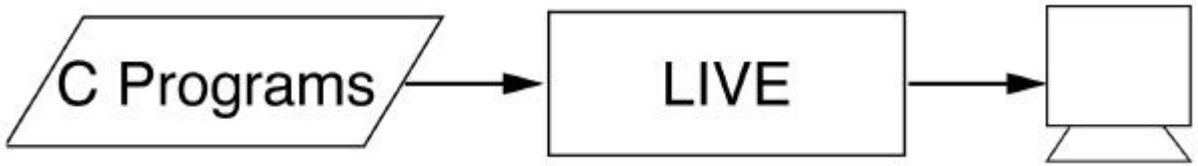
Execution Screen in HiTS



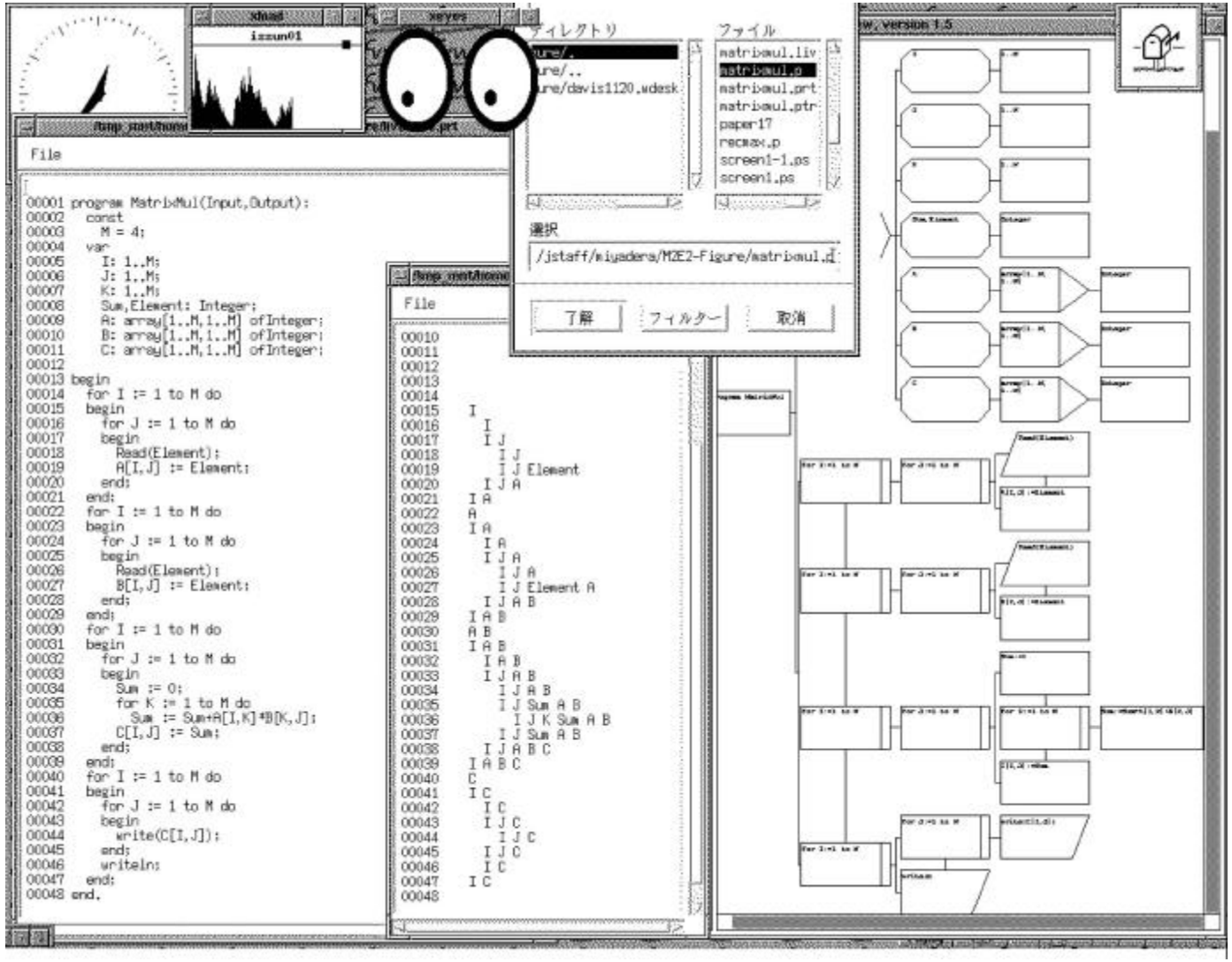
5.3 LIVE

Miyadera et.al. [9]

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

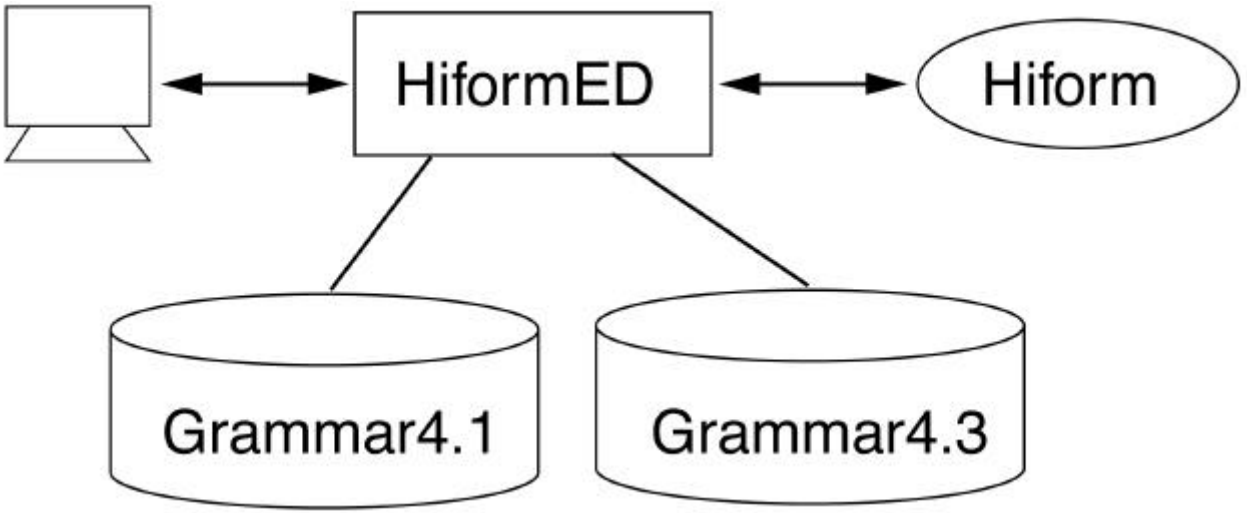


Execution Screen in LIVE

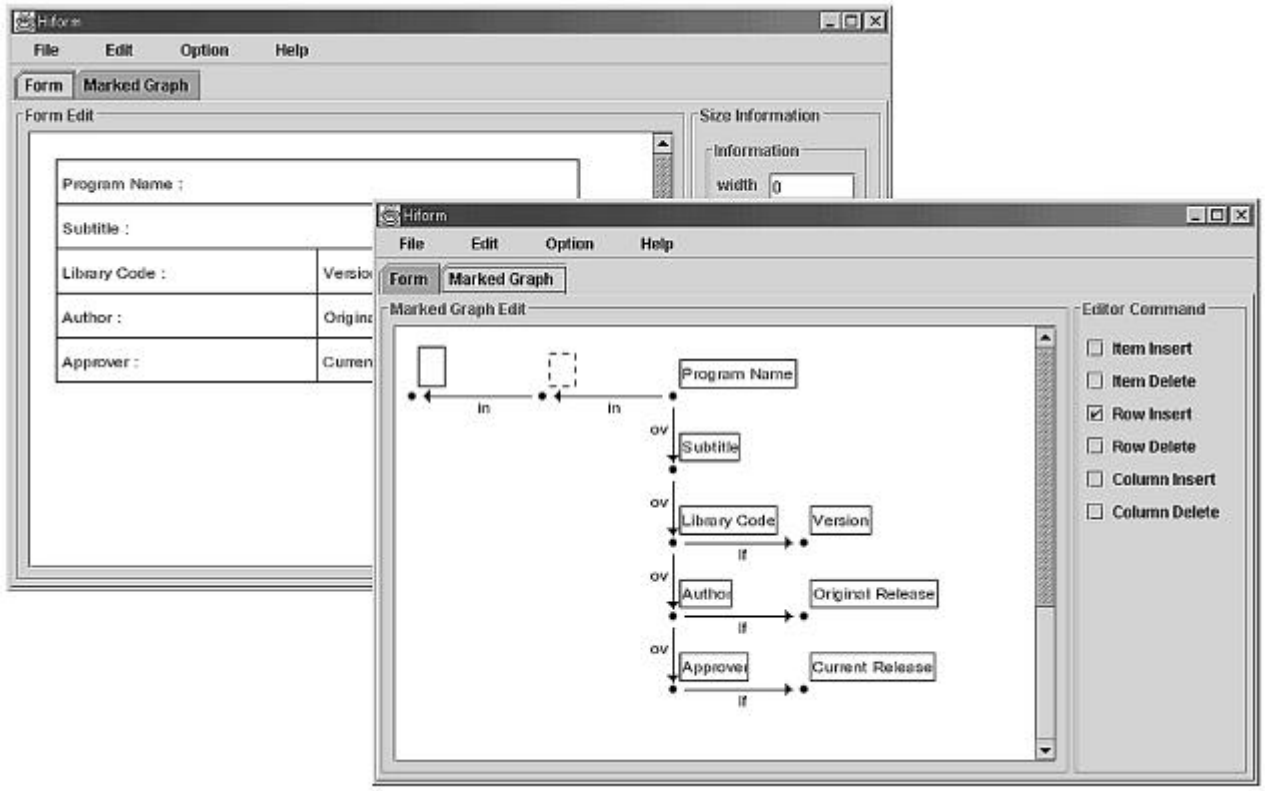


5.4 HiformED

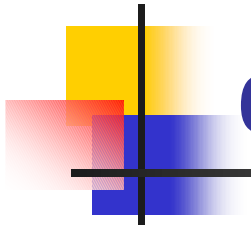
- Hiform Diagram Processing Component
- A Java application



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/