A Visual Programming Environment
Based on Graph Grammars and Tidy Graph Drawing
Y. Adachi, Y. Miyadera, K. Sugita
K. Tsuchida and T. Yaku
* Introduction : Purpose, Characteristics and Related Projects
* Hichart Tree Structured Diagram (TSD)
  1. Characteristics,
  2. Attribute Graph Grammar
* Hichart TSD Processing System
  1. Translator (HiTRANS),
  2. Editor (HiED)
* Hichart Tidy Drawing Algorithm (HiTD)
* Conclusion
Purpose and Characteristics

Motivation:
* Extension of Application Fields of Visualization.
* Need of Theoretical Background for Them.

Target:
Visualization of Programs.

Characteristics:
1. Using Tree Structured Diagram (TSD)
2. Applying Attribute Graph Grammar (AGG)
• Related Projects
  (1) IPSEN-PROGRESS targets program semantics,
  (2) PLAN2D targets nested graphs,
  (3) DiaGen, (4) GraphEd target general graphs.

• Components of Our System
  (1) Syntactical Editor (HiED),
  (2) Translator targeting Pascal/C/DXL (HiTRANS),
  (3) TSD Tidy Drawing Algorithms.
Fig. 1 Components of the System

HiED
Hichart Editor

Hichart TSD

HITRANS
Hichart Translator

C/Pascal Program

HITD
Hichart Tidy Drawing

Hichart Fair Print
program hanoimain

procedure hanoi

begin

param

output

begin

n
integer

if n=1 then

begin

a, b, c
char

T: writeln((a, b, c))

F

begin

hanoi(n-1, a, c, b)

end

end

begin

hanoi(3, 'A', 'B', 'C')

end

hanoi((n-1, a, c, b))

writeln(n, a, b)
2.1 Characteristics of Hichart TSD

• is a planar graph.
• shows both data structures and control flow simultaneously.
• is satisfying some graphical constraints.
  (e.g. (a) hierarchical alignment, (b) uniform gap between cells.)
2.2 Attribute Graph Grammar Used for Constructing This System

• AGG : (GGu, A, F)
  * GGu : underlying context-free graph
  * A : set of attributes (set of numerical values and functions.)
  * F : set of semantic rules (set of relations among attributes)
Inherited Attributes:
  x,y : x,y-coordinate of a cell
  RootX : x-coordinate of the root cell
  TopY : y-coordinate of the uppermost cell
  MinW, MinH : minimum width and minimum height of a cell
  GapX, GapY : spaces between adjacent cells in the vertical
direction and in the horizontal direction
  id : cell identifier

Synthesized Attributes:
  w,h : height and width of a cell
  up,low : upper space and lower space from the root cell
  of the subtree which are necessary to display
  the subtree
  cell : cell type
  string : string inside a cell
  lines : line connecting between cells
  ll : label of a cell
  cl : conditional label
  nc : cell number
If_statement(1)

Production

```
[if_statement] : =
    "if"
    <Boolean_Expression>
    "then"
    (in,out)
```

Semantic Rules

- $top(2) = top(0)$
- $bottom(0) = \max(bottom(1), bottom(2), bottom(3))$
- $top(3) = bottom(2) + \text{GapY}$
- $bottom(1) = y(1) + h(1)$
- $x(1) = x(0)$
- $nc(0) = 1 + nc(2) + nc(3)$
- $x(2) = x(1) + w(1) + \text{GapX}$
- $id(1) = id(0)$
- $x(3) = x(2)$
- $id(2) = id(1) + 1$
- $y(0) = y(1)$
- $id(3) = id(2) + nc(2)$
- $y(1) = (y(2) + y(3)) / 2$
- $w(1) = \text{MinW}$
- $h(1) = \text{get_height(}["if", <Boolean_expression>, "then"]}$
- $\text{cell}(1) = "exclusive_selection"$
- $\text{string}(1) = \text{get_str(}["if", <Boolean_expression>, "then"])$
- $\text{cl}(2) = "T:"
- $\text{cl}(3) = "F:"
- lines(1) = \{\text{get_line}(1,2), \text{getline}(1,3)\}$
HiTRANS: Translator of TSD

- **X2H**: Pascal/C/DXL to Hichart translator.
  - **Input**: Pascal/C/DXL source code,
  - **Output**: Hichart internal code and TSD.

- **H2X**: Hichart to Pascal/C/DXL translator.
  - **Input**: Hichart internal code,
  - **Output**: Pascal/C/DXL program code.
  
  *Rem. Hichart internal code is a text file.*
Main Phases of H2:

Parser: input: a Hichart internal code, output: a derivation tree for TSD.


Attribute-evaluator(B): input: a derivation tree, evaluation: attributes about drawing condition among cells.

Code-generator: generates program source codes.
Fig. 3 Parsing of a TSD
• **Feature**

  (a) Editing Commands
  (1) Generation, (2) Insertion, (3) Deletion.

  (b) Characteristics
  * Generation of Grammatically Correct TSDs.
  * Method Providing of Editing Command Based on the Production Rules of the HiAGG.
HiTD: Drawing Algorithm of TSD

HiTD: a program flowchart processing system that automatically generates tidy Hichart TSD using the theory of "tidy drawing problem".
Constraints of Layout

- Constraints:
  - (B1) planarity condition,
  - (B2,3) hierarchy alignment condition,
  - (B4) balancing condition of a sub-tree,
  - (B5) isomorphism for sub-trees,
  - (B6(k)) overlapping condition w.r.t a par.
  - (B7) other overlapping condition.
• **Conditions**

  \[ E_0 = B_1, B_2, B_3, B_4, B_5, B_6(0) \]
  \[ E_1 = B_1, B_2, B_3, B_4, B_5 \]
  \[ E_2 = B_1, B_2, B_3, B_4 \]
  \[ E_3(k) = B_1, B_2, B_3, B_4, B_5, B_6(k), B_7 \]
  \[ E_4 = B_1, B_2, B_3, B_4, B_5, B_7 \]

• **Algorithms**

  - **HiTD-Proc**: procedural algorithm
  - **HiTD-Cnst**: constraint based algorithm
  - **HiTD-Agg**: AGG based algorithm
• Development of a Visual Programming Environment Based on Graph Grammar.

• Definition of Complete AGG for Pascal/C/DXL.

• Development of a Syntactical Programming TSD Processing.

(1) HiED, (2) HiTRANS, (3) HiTD

• Implementation of a TSD Processing System Based on This AGG.