Syntactic Editing of Tabular Forms by Attribute edNCE Graph Grammars

6199M11 Kiyonobu Tomiyama
Supervisor Prof. Takeo Yaku

Abstract Tabular forms such as program specification forms [8] are naturally formalized by the attribute graphs [8], in which the attribute denotes locations of items and while the edge labels denotes relations between items. Documents of the tabular forms are represented by graph grammars (e.g., see [8]). Accordingly, a syntactic formalization of document editing provides the foundation for mechanical documentation. In this paper, first, we formalize syntax directed editing methods by extension of the notion of Cornell Program Synthesizer[2] to attribute edNCE graph grammars (cf.[4]). Next, we show the validity of our definition for editing a process under HNGG [8] using the confluence of HNGG.

Keywords Visual Programming, Software Development, Graph Grammars, Syntax Directed Editors

1 Introduction

Mechanical editing of tabular forms is one of the important issues in the software engineering methodology. The Cornell Program Synthesizer (CPS) is well-known and is often referred to as a structured and text-based editor which uses an attribute grammar successfully [2]. Tabular forms are represented by several different models (e.g., Pane represents them by [7]).

In this paper, we consider a programming documentation *Hiform* as an example of the tabular forms. Hiform document is a collection of 17 types of the tabular forms and includes all items defined in the guideline in ISO6592 [3],[5]. Those tabular forms are represented by graphs. Fig.1 illustrates a Hiform document and its corresponding graph. This graph is decided as follows: (1) A node label of graph shows an item of a tabular form. (2) An edge label shows relations between items. A mechanical processing of tabular forms supposed to be realized effectively by syntactic manipulation of graphs. In [8], the inner structure of each form in Hiform is defined by an attribute edNCE graph grammar.

The purpose of this paper is to exetend CPS mechanism to graph using results in [4][8] and to formalize a syntactic editing mechanism for graphs. Definition is made as to insertion and deletion in HNGG [8] so that two manipulations are validly executed by the confluence[6] of HNGG.

In Section 2, preliminary definitions are given. In Section 3, a formal definition for editing mechanisms are given, using composite production instance [4]. And we also show the validity of our definition using confluency of HNGG. Section 4, we investigate editor commands on HTGG. Section 5 is devoted for concluding remarks.

Program name : hanoi	
Subtitle :	
Library code : cs-2000-02	Version : 1.1
Author : K.Tomiyama	Original release :2000/6/10
Approver :	Current release :2000/10/1

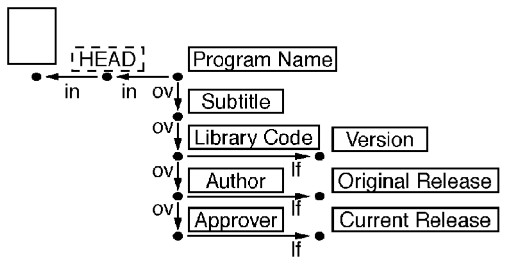

Figure 1: Tabular form in Hiform document and its corresponding graph

2 Preliminaries

We review an attribute edNCE graph grammar [8]. It is to be noted that the edNCE graph grammar allows for new edges to be established only between neighbouring nodes and embedded nodes as specified by connection instructions in the embedding process.

2.1 Attribute edNCE Graph Grammars [8]

We review an attribute graph grammar for the mechanical drawing. An *attribute NCE graph grammar* is as follows.

Definition [8] An attribute edNCE Graph Grammar is a 3-tuple $AGG =< G, Att, F >$ where
(1)$G = (\Sigma, \Delta, \Gamma, \Omega, P, S)$ is context-free edNCE graph grammar[6], called an *underlying graph grammar* of AGG. Each production p in P is denoted by $p : X \to (D, C)$. $Lab(D)$ denotes the set of all occurrences of the node symbols labeling the nodes in the graph D.

(2) Each node symbol $Y \in \Sigma$ of G has two disjoint finite sets $Inh(Y)$ and $Syn(Y)$ of *inherited* and *synthesized attributes*, respectively. We denote the set of all attributes of nonterminal node symbols Y by $Att(Y) = Inh(Y) \cup Syn(Y)$. $Att = \bigcup_{Y \in V} Att(Y)$ is called the *set of attributes* of AGG. We assume that $Inh(S) = \phi$. An attribute a of Y is denoted by $a(Y)$, and a set of possible values for a is denoted by $V(a)$.

(3) Associated with each production $p : X_0 \to (D, C) \in P$, there exists a set F_p of *semantic rules* which define all the attributes in $Syn(X_0) \cup \bigcup_{Y \in Lab(D)} Inh(Y)$. A semantic rule defining an attribute $a_0(X_{i0})$ has the form

$a_0(X_{i0}) := f(a_1(X_{i1}), \cdots, a_m(X_{im}))$, $0 \le i_j \le |Lab(D)|$, $X_{ij} \in Lab(D)$, $0 \le j \le m$. Here $|Lab(D)|$ denotes the cardinality of the set $Lab(D)$, and f is a mapping from $V(a_1(X_{i1}) \times \cdots \times a_m(X_{im}))$ into $V(a_0(X_{i0}))$. In this situation, we say that $a_0(X_{i0})$ depends on $a_j(X_{ij})$ for j, $0 \le j \le m$ in p. The set $F = \bigcup_{p \in P} F_p$ is called *the set of semantic rules of AGG.*

□

2.2 Compositions of Production Copies[4]

The composite representation of the production copies of an edNCE graph grammar is a theoretical and practical method for representing the graph-rewriting rules for embedding sub-graphs of desired structures into a graph.

Definition[4] Let $G = (\Sigma, \Delta, \Gamma, \Omega, P, S)$ be an edNCE graph grammar. Let $p_1 : X_1 \to (D_1, C_1)$ $(D_1 = (V_{D_1}, E_{D_1}, \lambda_{D_1}))$ and $p_2 : X_2 \to (D_2, C_2)$ $(X_2 = \lambda_{D_1}(u), D_2 = (V_{D_2}, E_{D_2}, \lambda_{D_2}))$ be production copies of G. If $u \subseteq V_{D_1}$, then a *composite production copy* (with a connection relation) $p : X_1 \to (D, C)$ is defined as follows:
D is a graph as $V_D = \{V_{D_1} - \{u\}\} \cup V_{D_2}$ about nodes.
$C = \{(\sigma, \beta/\gamma, \omega, d) \in C_1 | \omega \in V_{D_1} - \{u\}\}$
$\cup \{(\sigma, \beta/\delta, y, d) | \exists \gamma \in \Gamma, (\sigma, \beta/\gamma, u, d) \in C_1, (\sigma, \gamma/\delta, y, d) \in C_2\}$
The composite production copy p composed by p_1 and p_2, and denoted by $p_1 \circ p_2$.

□

The composite production copy $p_1 \circ p_2$ is also called *definable* in this case.

2.3 Confluence Property [6]

In general, the resulting graph of derivation based on an edNCE graph grammar depends on the order by which the production copies were applied.

The confluence property guarantees that the result of a derivation shall not depend on the order of the left applications of the production copies. Confluence is a very important property because it guarantees the validity of the left application of the composite production copies.

Definition [6] An edNCE graph grammar $G = (\Sigma, \Delta, \Gamma, \Omega, P, S)$ is *dynamically confluent* if the following holds for every sentential form H of G:
if $H \Rightarrow_{u_1,p_1} H_1 \Rightarrow_{u_2,p_2} H_{12}$ and $H \Rightarrow_{u_2,p_2} H_2 \Rightarrow_{u_1,p_1} H_{21}$ $(p_1, p_2 \in P)$ are (creative) derivation of G with $u_1, u_2 \in V_H$ and $u_1 \neq u_2$, then$H_{12} = H_{21}$.

□

2.4 HNGG [8][9]

In this section, we consider an attribute edNCE graph grammar. The grammar is called *Hiform Nested Graph Grammar* (HNGG). **HNGG** $= < G_N, A_N, F_N >$ that generates nested tabular forms called Hiform form. The following Fig.2 illustrates productions of HNGG.

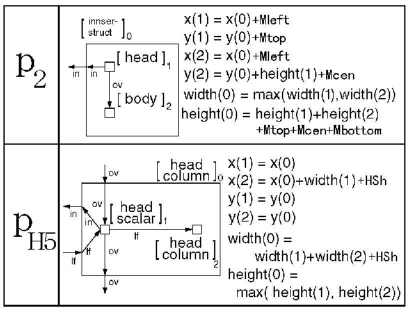

Figure 2: Productions with semantic rules of HNGG

3 Editing of Nested Tabular Form

In this section, we deal with a formal definition for editing manipulation using production instance of HNGG, and we also show the validity of our definition using confluency of HNGG.

3.1 Production Instance

We introduce editing manipulations in latter part. The editing manipulations are exactly defined by production instance as follows. In this part, we introduce production instance.

A *production instance* ("instance" for short) is a 3-tuple (ω, p_i, H'_{p_i}), where

1. $\omega \in V_{D_{i-1}}$ is a node removed during the derivation $D_{i-1} \Rightarrow_{p_i} D_i$.
2. $p_i : X_{p_i} \to (H_{p_i}, C_{p_i}) \in P$ 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 $D_{i-1} \overset{\omega H'_{p_i}}{\Rightarrow_{p_i}} D_i$ if D_{i-1} is directly derived D_i by applying the instance (ω, p_i, H'_{p_i}).
If there is a production sequence $p = (p_1, \cdots, p_n)$ and instance $(\omega_i, p_i, H'_{p_i})$ for each production p_i, an *instance sequence* is a sequence of $((\omega_1, p_1, H'_{p_1}), \cdots, (\omega_n, p_n, H'_{p_n}))$.

3.2 Syntactic Insertion

In this part, we define the syntactic insertion, and denote the flow of insertion manipulation. This manipulation is based on HNGG. Syntax directed editing is executed by sequences of production instances.

Definition For an derivation sequence $D_0 \overset{\omega_1 H'_{p_1}}{\Rightarrow_{p_1}} \cdots \overset{\omega_{i-1} H'_{p_{i-1}}}{\Rightarrow_{p_{i-1}}} D_{i-1} \overset{\omega H'_{p_i}}{\Rightarrow_{p_i}} D_i \overset{\omega_{i+1} H'_{p_{i+1}}}{\Rightarrow_{p+1}} \cdots \overset{\omega_n H'_{p_n}}{\Rightarrow_{p_n}} D_n$ $(p_j : X_{p_j} \to (H_{p_j}, C_{p_j}), 1 \le j \le n)$, we say that q is *insertable* (for p_i) if there is an instance (ω, q, H'_q) $(q :$

$X_q \rightarrow (H_q, C_q) \in P_N$) such that $D_{i-1} \overset{\omega H'_q}{\Rightarrow_q} Q$ and if there is a derivation sequence such that $D_{i-1} \overset{\omega H'_q}{\Rightarrow_q} Q \overset{\omega' H'_{p_i}}{\Rightarrow_{p_i}} D'_i \overset{\omega_{i+1} H'_{p_{i+1}}}{\Rightarrow_{p_{i+1}}} \cdots \overset{\omega_n H'_{p_n}}{\Rightarrow_{p_n}} D'_n$. Furthermore, if a production $q : X_q \rightarrow (H_q, C_q)$ is insertable for p_i, and any instance sequence applicable to D_i can be applied to D'_i, then q is *strictly insertable* (for p_i).

□

Inserting some instances into an instance sequence bring a new item into existence. That is, they correspond to a manipulation to insert a new item into a permissible place in a Hiform document.

Definition In the same manner as the editing by the instance for a production, we can further define *insertable by composite production copy.*

□

Definition A graph H' is *obtained by syntactic insertion* of a graph A at an edge x in a graph H.
1. A composite production copy q for the graph A and the 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. The graph H' is derived by the S.

□

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

□

3.3 Syntactic Addition

In this part, we define the syntactic addition, and denote the addition manipulation. This manipulation is based on HNGG.

Definition For an derivation sequence $D_0 \overset{\omega_1 H'_{p_1}}{\Rightarrow_{p_1}} \cdots \overset{\omega_{i-1} H'_{p_{i-1}}}{\Rightarrow_{p_{i-1}}} D_{i-1} \overset{\omega H'_{p_i}}{\Rightarrow_{p_i}} D_i \overset{\omega_{i+1} H'_{p_{i+1}}}{\Rightarrow_{p+1}} \cdots \overset{\omega_n H'_{p_n}}{\Rightarrow_{p_n}} D_n$ ($p_j : X_{p_j} \rightarrow (H_{p_j}, C_{p_j}), 1 \leq j \leq n$), we say that q is *addable* (for p_i) if there is an instance (ω, q, H'_q) ($q : X_q \rightarrow (H_q, C_q) \in P_N$, $H'_q = (V_{H'_q}, E_{H'_q}, \lambda_{H'_q})$) which satisfies the following.

1. There is an instance (ω, q, H'_q) such that $D_{i-1} \overset{\omega H'_q}{\Rightarrow_q} Q$ and a derivation sequence such that $D_{i-1} \overset{\omega H'_q}{\Rightarrow_q} Q \overset{\omega' H'_{p_{i+1}}}{\Rightarrow_{p_{i+1}}} D'_{i+1} \overset{\omega_{i+2} H'_{p_{i+2}}}{\Rightarrow_{p_{i+2}}} \cdots \overset{\omega_n H'_{p_n}}{\Rightarrow_{p_n}} D'_n$.
2. (a) $X_q = X_{p_i}$
   (b) $V_{H_q} = V_{H'_{p_i}} + \{u\}$
   (c) If f and g are isomorphic mappings such that
   $f : V_{H'_{p_i}} \rightarrow V_{H_{p_i}}$
   $g : V_{H_{p_i} + \{f(u)\}} \rightarrow V_{H_q}$,
   then $(\sigma, \beta/\gamma, y, d) = (\sigma, \beta/\gamma, g(y), d)$

□

Adding some instances into an instance sequence bring a new item into existence. That is, they correspond to a manipulation to add a new item into a permissible place in a Hiform document.

Definition In the same manner as the editing by the instance for a production, we can further define *addable by composite production copies.*

□

Definition A graph H' is *obtained by syntactic addition* of a graph A in a graph H.
1. A composite production copy q for the graph A exists.
2. There exists an instance i_q for q and an instance sequence i_H for H. An instance sequence S is obtained by addition of i_q into an instance sequence i_H.
3. The graph H' is derived by the S.

□

3.4 Syntactic Deletion of Item

We define here the deletion of inner most item in the form, that is a leaf node of the marked tree.

Definition For a derivation sequence $D_0 \overset{\omega_1 H'_{p_1}}{\Rightarrow_{p_1}} \cdots \overset{\omega_k H'_{p_k}}{\Rightarrow_{p_k}} F \overset{\omega H'_p}{\Rightarrow_p} D_p \overset{\omega_l H'_{p_l}}{\Rightarrow_{p_l}} \cdots \overset{\omega_n H'_{p_n}}{\Rightarrow_{p_n}} D_n$, let a graph D_p be the first graph of the instance sequence in which a node $u \in V_{D_p}$ is brought into existence and the node u is not rewritten by any production after that.
A production $p : X_p \rightarrow (H_p, C_p) \in P_N$ such that $H_p = (V_{H_p}, E_{H_p}, \lambda_{H_p})$ is *deletable* if one of the following Assumptions $1 - 3$ is met.
<Assumption 1>: For $p \in P_N$, there exists a production $p' : X_{p'} \rightarrow (H_{p'}, C_{p'}) \in P_N$ such that $H_{p'} = (V_{H_{p'}}, E_{H_{p'}}, \lambda_{H_{p'}})$ which satisfies the followings.
1. $X_{p'} = X_p$
2. $V_{H_{p'}} \equiv V_{H'_p} - \{u\}$
3. If f and g are isomorphic mappings such that
   $f : V_{H'_p} \rightarrow V_{H_p}$, $g : V_{H_p - \{f(u)\}} \rightarrow V_{H_{p'}}$,
   then $(\sigma, \beta/\gamma, y, d) = (\sigma, \beta/\gamma, g(y), d)$
<Assumption 2>: $V_{H'_p} = \{u, v\}$, $X_p = \lambda_{H'_p}(v)$
<Assumption 3>: $\omega_j \notin H'_p, l \leq j \leq n$

□

The deletion of an instance (ω, p, H'_p) from an instance sequence $((\omega_1, p_1, H'_{p_1})$,$\cdots$, $(\omega_n, p_n, H'_{p_n}))$ matches pruning or replacement of a derivation tree.

Definition A graph H' is *obtained by syntactic deletion* of a node A from a graph H.
$\overset{def}{\Leftrightarrow}$
1. A production q having a node A on the right hand side exists.
2. Let i_H is an instance sequence for H. Let i_q is an instance for q The i_q is deletable in i_H. An instance se-

quence S is obtained by deletion of i_q from i_H.
3. The graph H' is derived by the S.

$\square$

Proposition Let H be the graph obtaind from G by the deletion of nodes a and b in this ouder, in HNGG. Let H' be the graph obtaind from G by the deletion of nodes b and a in this order, in HNGG. Then, $H = H'$.

$\square$

4 Editing of Tessellation Tabular Forms

4.1 Tessellation Tabular Forms [8]

We introduce tessellation tabular forms based on a context-sensitive attribute edNCE graph grammar **HTGG** $= < G_T, A_T, F_T >$. Underlying graph grammar $G_T = (\Sigma_T, \Delta_T, \Gamma_T, \Omega_T, P_T, S_T)$ is context-sensitive edNCE graph grammar. Tessellation tabular forms have relations in every direction between the items. The following Fig.4 illustrates productions of tessellation tabular form.

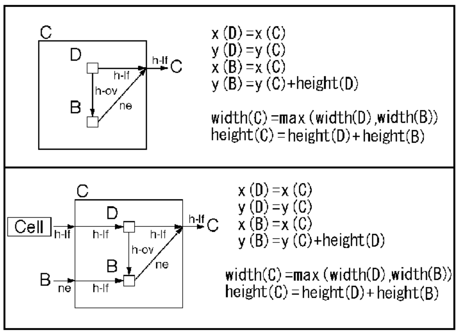

Fig.4 Productions of HTGG

4.2 Editor Commands of Tessellation Tabular Form

Definition Let $D_0 \overset{\omega_1 H'_{p_1}}{\Rightarrow_{p_1}} \cdots \overset{\omega_{i-1} H'_{p_{i-1}}}{\Rightarrow_{p_{i-1}}} D_{i-1} \overset{\omega H'_{p_i}}{\Rightarrow_{p_i}} D_i \overset{\omega_{i+1} H'_{p_{i+1}}}{\Rightarrow_{p_{i+1}}} \cdots \overset{\omega_n H'_{p_n}}{\Rightarrow_{p_n}} D_n$ ($p_j = X_{p_j} \rightarrow (D_{p_j}, C_{p_j}), 1 \leq j \leq n$) be a derivation sequence such that $x \in D_n$ is a head node of a selected line and D_i is a graph in which x appears for the first time. q is *insertable* for p_i if (1) there is a production sequence $q = (q_1 \cdots q_m)$ such that $D_{i-1} \overset{\omega H'_q}{\Rightarrow_q} Q$ and a derivation sequence such that $D_{i-1} \overset{\omega H'_q}{\Rightarrow_q} Q \overset{\omega'_i H'_{p_i}}{\Rightarrow_{p_i}} D'_i \overset{\omega_{i+1} H'_{p_{i+1}}}{\Rightarrow_{p_{i+1}}} \cdots \overset{\omega_n H'_{p_n}}{\Rightarrow_{p_n}} D'_n$ and (2) it is possible to apply any derivation sequences for D_i, then it is also possible to apply them for D'_i. *Insertion* is similarly defined as in **HNGG**.

$\square$

Definition It prepares $q = (p_{17}, p_{18}, (p_{19}, p_{20})^k, p_{21}, p_{22}, p_{14}, p_{15}^k, p_{16})$ as the *insertion command.*

$\square$

5 Conclusion

We proposed editing method for tabular forms, based on attribute edNCE graph grammar of tabular forms with a homogenous cell size. Our editing method also includes attribute rules for mechanical drawing. By using this editing method, we can exactly edit valid tabular forms defined by NCE graph grammar.

Furthermore we are now developing a tabular form editor system by using this approach.

Acknowledgment

We thank Mr. S. Kanai's advise in the course of preparing the manuscript.
We also thanks to Tomokazu Arita, Shu-ichi Nakagawa and Kenji Ruise for valuable discussions.

References

[1] Reinhold Franck, A Class of Linearly Parsable Graph Grammars, Acta Infomatica 10, 175-201 (1978)

[2] Tim Teitelbaum and Thomas Reps, The Cornell Program Synthesizer: A Syntax-Directed Programming Environment, *Comm. ACM*, Vol.24, 563-573, (1981).

[3] ISO6592-1985, Guidelines for the Documentation of Computer-Based Application Systems, (1985).

[4] Y.Adachi, K.Anzai, et al. Hierarchical Program Diagram Editor Based on Attribute Graph Grammar, *Proc. COMPSAC*'96, 205-213(1996).

[5] K. Sugita, Y. Adachi, Y. Miyadera, K. Tsuchida and T. Yaku, Advanced Software Mechanisms for Computer-Aided Instruction in Information Literacy, *APEC-CIL*'97, (1997).

[6] Grsegorz Rozenberg (Ed.), Handbook of Graph Grammar and Computing by Graph Transformation,World Scientific Publishing(1997).

[7] John F. Pane, Brad A. Myers, Tabular and Textual Methods for Selecting Objects from a Group, *Proc. 2000 IEEE Symp. on Visual Language*, 157-164, (2000).

[8] T. Arita, K. Tomiyama, T. Yaku, Y. Miyadera, K. Sugita, K. Tsuchida, Syntactic Processing of Diagrams by Graph Grammars, *Proc. IFIP WCC ICS 2000*,145-151 (2000).

[9] T. Arita et al., A Precedence Attribute NCE Graph Grammar for Hiform, Http://www.hichart.org/keyaki/archive/HC00-001