表編集のアルゴリズム

本橋友江
関東学院大学工学部

谷聖一
日本大学文理学部

土田賢省
東洋大学工学部

夜久竹夫
日本大学文理学部

Algorithms for Table Transformation

Tomoe Motohashi, Kanto Gakuin University
Sei'ichi Tani, Nihon University
Kensei Tsuchida, Toyo University
and
Takeo Yaku, Nihon University

Abstract:

Tables with heterogeneous cells are commonly used in computer human interface and documentation. We proposed an attribute multi edge graph representation for tables that considers editing and drawing in [7]. In this paper, we give algorithms for basic operations in table editing.

We provide a cell unification algorithm that runs in $O(1)$ time. We also provide a column insertion algorithm that runs in $O(n+m)$, for $n \times m$ heterogenious tables. It is noted that the column insertion algorithm runs in $O(\sqrt{N})$ time for $N = n \times n$ cell square tables. while known methods require $O(N)$ time

*

Keywords: Data structures and algorithms, graphs, table interface

1 Introduction

In this paper, we deal with the representation of tables while considering editing and drawing. Several representation methods have been proposed for tables: rectangular duals of planar graphs [1], and quadtrees [3]. Although, we do not know which representation method is used in present table processing systems.

In this paper, we considers another graph representation [7] for tables, in which the table editing is executed efficiently. For drawing and editing problems, see [4, 5].

*This paper partly appeared in our previous study Tomoe Motohashi, Kensei Tsuchida and T. Yaku, Algorithm on attribute graphs for table editing, The 3rd Hungarian-Japanese Sumpos. Discrete Math. & Its Appl. and [6].

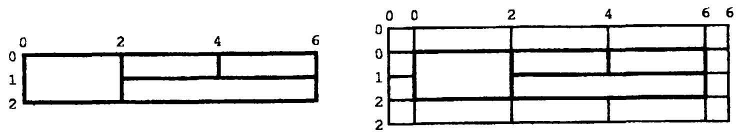

Figure 1: A: A Tabular Diagram D_1 B: A Tabular Diagram D_{1p}

In Section 2, we propose a representation of tables by an attribute multi edge graph. Several properties of the graphs are shown. In Section 3, we show several algorithms that execute table editing based on the representation. We provide algorithms for unifying cells, changing column width, and insertion column. Section 4 provides conclusions.

2 ATTRIBUTE GRAPHS FOR TABLES

We described several definitions concerning tables in [7]. We represent a table by a certain type of tabular diagram satisfying Condition 2.1 in [7]. That is, the tabular diagrams have perimeter cells.

Example 2.1 Figure 1A illustrates a tabular diagram $D_1 = (T_1, P_1, g_1)$, where $P_1 = \{\{(1,1),(2,1)\},\{(1,2)\},\{(1,3)\},\{(2,2),(2,3)\}\}$ is a partition over a (2,3)−table T_1. A grid g_1 is defined by $g_{1row}(0) = 0, g_{1row}(1) = 1$, $g_{1row}(2) = 2$, and $g_{1column}(0) = 0, g_{1column}(1) = 2$, $g_{1column}(2) = 4$, and $g_{1column}(3) = 6$. For a cell $c = \{(2,2),(2,3)\}$, we define the north wall $nw(c) = g_{1row}(1) = 1$, south wall $sw(c) = g_{1row}(2) = 2$, east wall $ew(c) = g_{1column}(3) = 6$, and west wall $ww(c) = g_{1column}(1) = 2$. Figure 1B illustrates a tabular diagram D_{1p} with perimeter cells. For the table drawing, it corresponds to D_1.

Now, we introduce an attribute graph. Then, we show how to represent a tabular diagram as an attribute graph.

Definition 2.1 An *attribute graph* is a 6−tuple $G = (V, E, L, \lambda, A, \alpha)$, where (V, E) is a *multi-edge undirected graph*, L is the set of *labels* for edges, $\lambda : E \rightarrow L$ is the *label function*, A is the set of *attributes*, and $\alpha : V \rightarrow A$ is the *attribute map*.

A tabular diagram $D = (T, P, g)$ is *represented* as an attribute graph $G_D = (V_D, E_D, L, \lambda_D, A, \alpha_D)$, where V_D is identified by a partition P (we denote a node corresponding to a cell c in P by v_c, we call v_c a *perimeter node* (resp. *inner node*) if c is a perimeter cell (resp. inner cell)), E_D is defined by Rules 1-4, $L = \{enw, esw, eew, eww\}$, $\lambda_D : E_D \rightarrow L$ is defined by Rules 1-4, $A = R^4$, and $\alpha_D : V_D \rightarrow R^4$ are defined by $\alpha_D(v_c) = (nw(c), sw(c), ew(c), ww(c))$.

Rule 1 If $nw(c) = nw(d)$ and there is no cell between c and d having an equal north wall, then $[v_c, v_d]$ is in E_D and $\lambda_D[v_c, v_d] = enw$. In this case, $[v_c, v_d]$ is called a *north wall edge*. Similarly, **Rules 2**, **3** and **4** define the *south wall*, *east wall*, and *west wall edges*, respectively.

An attribute graph G_D is called a *tessellation graph*. Note that the degree $\#v$ of each node v in G_D is at most 8. The location vlues of the inner cells are evaluated from the location values of perimeter cells and linked edges. So, we assume in the latter part of this paper, that the α_D values of the inner cells are null.

Note that we consider tabular diagrams with perimeter cells. Then,

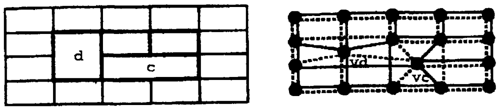

Figure 2: Tabular Diagram and Its Corresponding Tessellation Graph

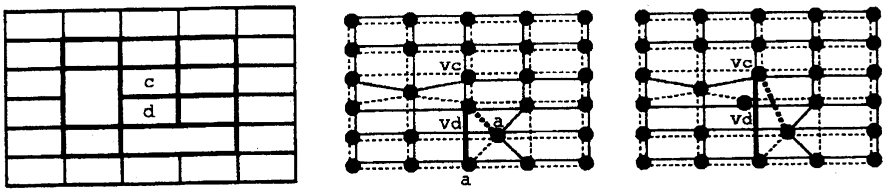

Figure 3: Change of Vertical Edges of v_d

Proposition 2.1 *Let G_D be a tessellation graph for a tabular diagram D of an (n,m)−table. Let k be the number of inner cells in G_D.* For the number $\#E_D$ of edges in G_D, we have $2\#E_D = 6(2n-4) + 6(2m-4) + 8k + 16$.

3 ALGORITHMS

This section provides algorithms for tessellation graphs. The following algorithm execute unification of two adjacent inner cells in the tabular diagram.

ALGORITHM UNIFYCELLS(G_D, v_c, v_d, G_E)
INPUT
$G_D = (V_D, E_D, L, \lambda_D, A, \alpha_D)$: a tessellation graph for a tabular diagram D, v_c : a node in G_D corresponding to an inner cell c, v_d : a node in G_D corresponding to an inner cell d which is adjacent to the south wall of c that is $ww(c) = ww(d), ew(c) = ew(d), sw(c) = nw(d)$.
OUTPUT
$G_E = (V_E, E_E, L, \lambda_E, A, \alpha_E)$: a tessellation graph for a tabular diagram E, where E is obtaind from D by the unification of cells c and d to c.
METHOD
begin
Initially let $G_E \leftarrow G_D$;
/* change of vertical edges concerning to d */
delete two vertical edges between v_d and $a(\neq v_c)$ from E_E ;
add edges between v_c and a to E_E ;
put $\lambda_E[v_c, a] \leftarrow \lambda_D[v_d, a]$;
delete two vertical edges between v_c and v_d from E_E ; (See Fig.3)
/* change of south wall edges concerning to c */
choose two nodes f and $f'(f \neq f')$ such that $\lambda_D[f, v_c] = \lambda_D[v_c, f'] = esw$;
delete south wall edges $[f, v_c]$ and $[v_c, f']$;
add an edge $[f, f']$ to E_E and put $\lambda_E[f, f'] \leftarrow esw$;
/* change of south wall edge concerning to d */

choose two nodes h and $h'(h \neq h')$ such that $\lambda_D[h, v_d] = \lambda_D[v_d, h'] = esw$;
delete south wall edges $[h, v_d]$ and $[v_d, h']$;
add $[h, v_c]$ and $[v_c, h']$ to E_E and put $\lambda_E[h, v_c] \leftarrow esw, \lambda[v_c, h'] \leftarrow esw$;
/* change of north wall edges concerning to d */
choose two nodes g and $g'(g \neq g')$ such that $\lambda_D[g, v_d] = \lambda_D[v_d, g'] = enw$;
delete north wall edges $[g, v_d]$ and $[v_d, g']$ from E_E ;
add an edge $[g, g']$ to E_E and put $\lambda_E[g, g'] \leftarrow enw$;
delete a node d
end.

Theorem 3.1 *Let D be a tabular diagram, and c be a cell in D. Suppose that there is an adjacent cell d at south side in D such that $ew(c) = ew(d)$, $ww(c) = ww(d)$ and $sw(c) = nw(d)$. Let E be a tabular diagram obtained from D by the unification of cells c and d to c. Let G_D and G_E be the tessellation graphs for D and E, respectively. Then G_E is obtained from G_D in constant time.*

Theorem 3.2 *Let D be a tabular diagram, and c be an inner cell in D. Let δ be a movement value. Suppose $\Delta + \delta > 0$, where $\Delta > 0$ is the width of a perimeter cell in the column which has equal east wall to c. Let E be a tabular diagram obtained from D by the changing width using δ of cells that have the equal east wall for c. Let G_D and G_E be the tessellation graphs for D and E, respectively. Then G_E is obtained from G_D in $O(n + m)$ time by the algorithm* ChangeColumnWidth *[6], where n is the number of rows in D.*

The following algorithm executes insertion of a column at the west side of a focused cell into the tabular diagram.

ALGORITHM InsertColumn(G_D, v_c, G_E)
INPUT
G_D : a tessellation graph for a tabular diagram $D = (T, P, g)$, v_c : a node in G_D corresponding to a cell c, where the cell, that is adjacently located at the west-side of c, exists.
OUTPUT
G_E : a tessellation graph for E, where E is a tabular diagram obtained from D by insertion of a column with width δ at the west side of c, where δ is the width of a perimeter cell in the column including c.
METHOD
begin
Initially, put $G_E \leftarrow G_D$;
traverse upward through the west wall edges from v_c until a perimeter node v_0 (see Fig.5) ;
let δ be the width of the cell corresponding to v_0 ;
add a node u_0 ;
put $i \leftarrow 0$;
/* insert a column */
while a node v_i is not the lowermost node **do begin**
let w_i be an adjacently west-side node linked to v_i by a north wall edge ;
delete the north wall edge between w_i and v_i ;
add a north wall edge between w_i and u_i ;
deform G_E similarly for a south wall edge ;

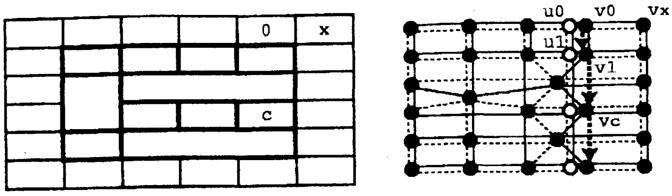

Figure 4: Insertion of the Column at the West-Side of c

add a north wall edge and south wall edge between u_i and v_i ;
let v_{i+1} be a lower node linked to v_i by a west wall edge ;
add a node u_{i+1} ;
add a west wall edge and east wall edge between u_i and u_{i+1} ;
$i \leftarrow i+1$
end ; (see Fig.6)
deform G_E for the lowermost node v_i, similarly for the noth and south wall edge;
/* the existing column shifts to the east */
let u_0 be the uppermost node in u_i's ;
let v_x be adjacently west-side node linked to the node in the northeast corner in G_E ;
put $G_{E_0} \leftarrow G_E$;
CHANGECOLUMNWIDTH$(G_{E_0}, v_x, \delta, G_E)$;
put $G_{E_0} \leftarrow G_E$;
let x_0 be a node adjacently west-side of v_x ;
put $i \leftarrow 0$;
while $ww(u_0) \leq ww(x_i)$ **do begin**
MOVEEASTWALL$(G_{E_i}, x_i, \delta, G_{E_{i+1}})$;
let x_{i+1} be an adjacently west-side node linked to x_i by a north wall edge ;
put $i \leftarrow i+1$;
end ;
put $G_E \leftarrow G_{E_i}$
end.

Theorem 3.3 *Let D be a tabular diagram, and c be a cell in D. Suppose that E is the tabular diagram obtained from D by insertion of a column with width δ at the west side of the column including c, where δ is the width of a perimeter cell of the column including c. Let G_D and G_E be the tessellation graphs for D and E, respectively. Then G_E is obtained from G_D in $O(n+m)$ time, where n and m are the number of rows and columns in D, respectively.*

It is noted that the column insertion algorithm runs in $O(\sqrt{N})$ time for $N = n \times n$ cell square diagrams. while known methods require $O(N)$ time. The following table illustrates the features of representation methods for N cell square tables with respect to the column insertion.

Model	Node degrees	Cell to node relation	Cell visits	Complexity
Quadtrees [1]	at most 5	one 'block' to one node	at most N	$O(N)$
Rec-tangular dual graphs [2]	at most $4N$	one cell to one node	at most N	$O(N)$
Tessellation graphs	at most 8	one cell to one node	at most $2\sqrt{N}$	$O(\sqrt{N})$

4 CONCLUSION

We introduced attribute graphs and algorithms for table drawing and editing. We note that, we have determined the necessary and sufficient condition where an attribute graph represents a tabular diagram by a graph grammar. [9]. We are designing a processing system for table editing based on our model in [8].

The authors thank to Prof. Tominaga of Tokyo University of Technology for valuable discussion with him.

References

[1] K. Kozminski, E. Kinnen, Rectangular Duals of Planar Graphs, *Networks 15* (1985) 145-157.

[2] Franz J. Brandenburg, Designing Graph Drawings by Layout Graph Grammers, *Proc. Graph Drawing '94, LNCS 894* (1994) 416-427.

[3] M. de Berg, M.van Kreveld, M. Overmats and O. Schwarzkoph , Computational Geometry - Algorithms and Applications, *Springer* (1997)

[4] T. Arita, K. Tsuchida, T. Yaku et. al. , Syntactic proccessing of diagrams by graph grammers, *Proc. IFIP World Computer Congress ICS 2000* (2000) 145-151.

[5] A. Amano, N. Asada, T. Motoyama, T. Sumiyoshi, K. Suzuki, Table Form Document Systhesis by Grammar-Based Structure Analysis, *6-th Internal Conference on Document Analysis and Recognition (ICDAR)* (2001) 533-537

[6] T. Motohashi, K. Tsuchida, T. Yaku, Attribute Graphs and Their Algorithms for Table Interface, *TECHNICAL REPORT OF IEICE SS2002-1* (2002).

[7] T. Motohashi, K. Tsuchida, T. Yaku, Proc. Attribute Graphs for Tables and Their Algorithms, *fose 2002* (2002), (井上克郎編、ソフトウェア工学の基礎 I X、近代科学社)、183 - 186.

[8] T. Kirishima, T. Motohashi, K. Tsuchida, T. Yaku, Attribute Graph for Table and Their Applications, *Proc. IASTED International Conference on Software Engineering and Applications SEA 2002* (2002), 317 - 322.

[9] T. Kirishima, T. Arita, T. Motohashi, K. Tsuchida and T. Yaku, Syntax for Tables, *Proc. IASTED International Conference on Applied Informatics AI 2003* (2003), *to appear*.