An Implementation of the 16-ary Grid Graphs for the Multiply Layered Rectangular Dissections

Koichi Anada¹, Koushi Anzai², Shinji Koka³, and Takeo Yaku³  
¹Waseda Research Institute for Science and Engineering, Waseda University, Tokyo, Japan  
²Department Economics, Kanto Gakuen University, Gunma, Japan  
³Department Computer Science and System Analysis, Nihon University, Tokyo, Japan

Abstract - Heterogeneous rectangular dissections are frequently used in information processing such as multiple paged books in spread sheet languages and multiple layered image data. In previous studies, a hexadecimal grid graph model was proposed for multiple layered rectangular dissections and certain algorithms were provided.

In this paper, we propose a 32-ary list structure to implement their algorithms for the hexadecimal grid graph model. The list structure has limited number of fields in a record, so the computation time is low.

Furthermore, it has one record for each node in the given hexadecimal grid, and 48 fields for each record. We also show a data format of the whole structure of the list.

Keywords: modeling of spreadsheets, hexadecimal grids, list structures, multiple layered rectangular dissections

1 Introduction

Heterogeneous rectangular dissections are frequently used in visualization of information such as multiple paged books in spread sheet languages and multiple layered image data. In previous studies, a hexadecimal grid graph model was proposed for multiple layered rectangular dissections and certain algorithms were provided [2, 3]. The formalization of business documents as in Figure 1 has become an important subject with the progress of e-commerce and e-government (see, e.g. [1]). In order to formalize financial statements, we have to specify the spatial order of items, and specify calculation methods of categorized items. A context sensitive graph grammar is proposed in [6] that specify financial statements.

In section III, we propose a 32-ary list structure to implement their algorithms for the hexadecimal grid graph model. The list structure has limited number of fields in a record, so the computation time is low. Furthermore, it has one record for each node in the given hexadecimal grid, and 48 fields for each record. We also show a data format of whole structure of the list in section IV.

2 Multilayer rectangular dissections and hexadecimal grids

Figure 1 illustrates a k-layered multilayer rectangular dissection D (left) and its corresponding hexadeci-grid G_D (right).

In a hexadeci-grid, two nodes are horizontally linked if they correspond nearest cells with the ruled line in common, and two nodes are vertically linked if they correspond nearest cells with their corner in common. Figure 2 shows the links around an inner node in a hexadeci-grid, and corresponding cells in a k-layered multiply layered rectangular dissections.

3 32-ary list structures for hexadecimal grids

In this section, we represent the hexadeci-grids by list structures. We assign one record in the list structure for one node in the hexadeci-grid. The structure of each record is illustrated as Figure 3.
Each record in H4CODE list in represented by the following 48 field data. The H4CODE list is represented by a repetition of those 48 field data.

Table 2. A data for each record in an H4CODE list.

<table>
<thead>
<tr>
<th>Field Number</th>
<th>Name</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>node id</td>
<td>id for each node</td>
</tr>
<tr>
<td>02</td>
<td>cell type</td>
<td>perimeter / inner</td>
</tr>
<tr>
<td>03</td>
<td>new_right</td>
<td>the link to the nearest cell with the ruled line in common</td>
</tr>
<tr>
<td>04</td>
<td>new_left</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>42 - 48</td>
<td>Not Used</td>
<td></td>
</tr>
</tbody>
</table>

5 Concluding remarks

In this paper, we designed a list structure called an H4CODE list for hexadecimal grids, and also designed a data format called H4CODE. An H4CODE list consists of records as illustrated in Figure 3. And, the H4CODE format consists of a header block with 8 fields and a list block with 48 fields. We are implementing several processing systems such as tabular form editing systems, and terrain map systems using H4CODE (cf. [3, 4]). We would like to thank Profs. Goro Akagi, Kensei Tsuchida, and Mr. Kenshi Nomaki for their valuable suggestions.

6 References


