数理解析研究所講究録
1489巻 2006年 78-84

Syntactic Characterization of Two-Dimensional Grid Graphs by a Context-Sensitive Graph Grammar *

桜美林大学 文学部 言語コミュニケーション学科 有田 友和 (Tomokazu ARITA)
Department of Languages and Information Studies, Obirin University

東洋大学 工学部 情報工学科 土田 賢省 (Kensei TSUCHIDA)
Department of Information and Computer Sciences, Toyo University

日本大学 文理学部 情報システム解析学科 夜久 竹夫 (Takeo YAKU)
Department of Computer Science and System Analysis, Nihon University

Abstract

Graph grammar can characterize several type of graphs. Vigna and Ghezzi showed that the language of grid graphs could not be constructed by their context-free graph grammars [4]. In this paper, we give a graph grammar for two-dimensional grid graphs. The graph gramamr for grid graphs is context-sensitive.
keywords: context-sensitive graph grammar, grid graphs.

1 Introduction

Our interest is in a graph grammar for two-dimensional grid graphs.

Graph grammar has succeeded in characterizing graph structures and generating several types of graphs [3, 9, 8]. In [5] and [6], there are graph grammars that can derive ladders. Vigna and Ghezzi showed that a language of grid graphs could not be constructed using their context-free graph grammars [4]. In [1], Pavlidis also described that the set of grid graphs could not be generated using their context-free graph grammars.

On the other hand, Burosch and Laborde characterized grid graphs using the products of paths and showed that two-dimensional grids can be recognized in linear time by applying a combinatorial algorithm [7].

Graph grammar characterization for grid graphs possibly derives characterization of grid graphs with multiple labels, such as business document tables [10] or symbol tables in program documents.

In this paper, We show that there is a context-sensitive graph grammar for two-dimensional grid graphs. This grammar derives grid graphs by synchronizing using state propagation among nodes in graph rewriting rules. Label propagation is based on state propagation of cellular automata (see e.g., [2]).

2 Notations and Definitions

First, we begin by reviewing some graph notations. For this paper, we consider directed graphs with node and edge labels. We also consider undirected graphs by extending the directed graphs.

Definition 1 [9] Let Σ be an alphabet of node labels and Γ an alphabet of edge labels. A ***graph*** over Σ and Γ is $G = (V, E, \psi)$, where V is the finite set of ***nodes***, $E \subseteq \{(v, l, w) | v, w \in V, l \in \Gamma\}$ is the set of ***edges***, and $\psi : V \to \Sigma$ is the ***labeling*** function.

*The result in this paper appeared in T. Arita, K. Tsuchida, and T. Yaku, "Syntactic Characterization of the Two-Dimensional Grid Graphs", IEICE Transactions on Information and Systems, vol.E89-D, no.2, pp.771–778, February 2006 [13]. and part of the result in this paper appeared in Section 4 of T. Arita et al., "Syntactic Processing of Diagrams by Graph Grammars", the Proceedings of 16th IFIP World Computer Congress 2000 ICS2000, pp.145-151, 2000. [12].

In this paper, we always assume that G is connected. Graph $G = (V, E, \psi)$ is an undirected graph if, for every $(v, l, w) \in E$, there is also $(w, l, v) \in E$ [9].

Definition 2 [7] A (undirected) graph is called *a (two-dimensional) grid graph* iff the graph consists of the product of two path graphs [7]. Let P_{n1} and P_{n2} be path graphs, then $P_{n1} \Box P_{n2}$ denotes the product of P_{n1} and P_{n2}. If P_{n1} is a path graph with $n1$ nodes and P_{n2} is a path graph with $n2$ nodes, then $T_{n1,n2}$ denotes a grid graph for the product of the P_{n1} and P_{n2}.

Figure 1 shows a grid graph $T_{3,4}$.

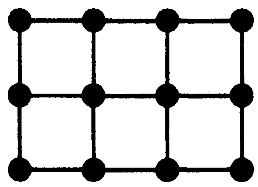

Figure 1: A grid graph.

We now consider that two graphs are "isomorphic." Two graphs $G_1 = (V_1, E_1, \psi_1)$ and $G_2 = (V_2, E_2, \psi_2)$ are *isomorphic* if there is a bijection f: $V_1 \to V_2$ such that $E_2 = \{ (f(v), l, f(w)) \mid (v, l, w) \in E_1 \}$ and, for all $v \in V_1$, $\psi_2(f(v)) = \psi_1(v)$ [9].

3 Context-Sensitive Graph Grammars

Next we consider graph transformations. Graphs are transformed based on production rules. We deal with the following productions, in which one production replaces a subgraph in one graph with other graph.

In this section, we modify edNCE graph grammars [9] and introduce context-sensitive graph grammars. The left-hand side of the production in the edNCE graph grammar is defined using a single node label. But the left-hand side of the production in our graph grammar is defined using a connected graph.

Definition 3 (cf. [9]) A *production* p is of the form $M \to (D, C)$, where M is a graph over Σ and Γ, D is a graph over Σ and Γ called an *embeded graph*, and $C \subseteq \Sigma \times \Gamma \times V_M \times \Gamma \times V_D \times \{in, out\}$, called a *connection relation*, is a set of *connection instructions*.

A production p: $M \to (D, C)$ removes a subgraph M' of its applied graph G, such that M and M' are isomorphic, and replaces M' with D', such that D' and D are isomorphic and the set of nodes in G and D' are pairwise disjoint. Then the connection instructions of C replace neighboring edges around M'. A connection instruction $(\alpha, \beta, x, \gamma, y, out)$ of C means that if there was an edge labeled β from the node x in G for which (D, C) is substituted to a node w with label α, then the embedding process will establish an edge labeled γ from y to w. Similarly, a connection instruction $(\alpha, \beta, x, \gamma, y, in)$ of C means that if there was an edge labeled β from a node w with label α to the x in G for which (D, C) is substituted, then the embedding process will establish an edge labeled γ from w to y.

That is, let $G = (V_G, E_G, \psi_G)$ and $H = (V_H, E_H, \psi_H)$ be graphs on Σ and Γ. Let p: $M \to (D, C)$ be a production such that V_G and V_D are pairwise disjoint. Then $G \Rightarrow_p H$ holds for p if there is a subgraph M' of G such that M' and M are isomorphic. That is we obtain H from G as follows: $V_H = (V_G - V_{M'}) \bigcup V_D$, $E_H = \{(x, l, y) \mid x, y \in V_G - V_{M'} \} \bigcup E_D \bigcup \{(w, l, x) \mid \exists m \in \Gamma, \exists v \in V_{M'}: (w, m, v) \in E_G, (\psi_G(w), m, v, l, x, in) \in C\} \bigcup \{(x, l, w) \mid \exists m \in \Gamma, \exists v \in V_{M'}: (v, m, w) \in E_G, (\psi_G(w), m, v, l, x, out) \in C\}$, $\psi_H(x) = \psi_G(x)$ if $x \in (V_G - V_{M'})$, and $\psi_H(x) = \psi_D(x)$ if $x \in V_D$ (cf. [9]).

Definition 4 ([9]) A *graph grammar* is $GG = (\Sigma, \Delta, \Gamma, \Omega, P, S)$, where Σ is the finite set of *node labels*, $\Delta \subseteq \Sigma$ is the set of *terminal node labels*, Γ is the finite set of *edge labels*, $\Omega \subseteq \Gamma$ is the set of *final edge labels*, P is the set of *productions*, and S is the *start graph*.

We call *context-sensitive* graph grammar, in which the left-hand side of a production is a graph.

For productions $M_1 \to (D_1, C_1)$ and $M_2 \to (D_2, C_2)$, Two connection relations C_1 and C_2 are called isomorphic if there are isomorphism f from the node set of M_1 to the node set of M_2 and f' from the node set of D_1 to the node set of D_2 such that $C_2 = \{ (\sigma, \beta, f(v), \gamma, f'(x), d) \mid (\sigma, \beta, v, \gamma, x, d) \in C_1 \}$.

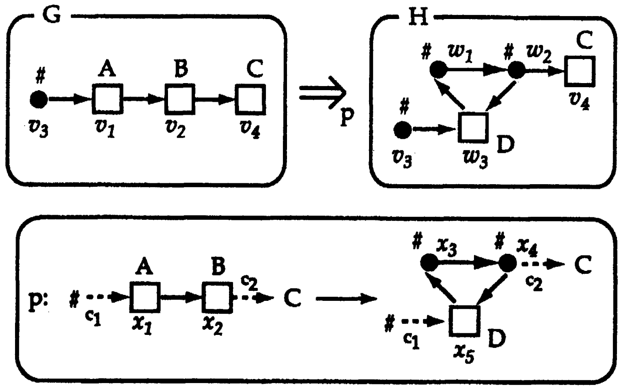

Figure 2: An example of an application of a production.

Two productions $M_1 \rightarrow (D_1, C_1)$ and $M_2 \rightarrow (D_2, C_2)$ are called isomorphic if M_1 and M_2 are isomorphic graphs, D_1 and D_2 are isomorphic graphs, and C_1 and C_2 are isomorphic.

Let $GG = (\Sigma, \Delta, \Gamma, \Omega, P, S)$ be a graph grammar. Let G and H be graphs on Σ and Γ and let p: $M \rightarrow (D, C)$ be a production in P. Then $G \Rightarrow_p H$ is called a ***derivation step***, and a sequence of derivation steps is called a ***derivation*** (cf. [9]).

We assume that P does not contain distinct isomorphic productions in this grammar.

Figure 2 shows an example of a production and a derivation of a graph. Node labels A, B, C, D are non-terminal, and label # is terminal. Every edge in the graphs shown in Figure 2 is unlabeled. Furthermore, a box is a nonterminal node, a black circle is a terminal node, an arrow is a directed edge, and a dotted arrow is part of a connection instruction. We use edge label # for unlabeled edges. Production p is $M \rightarrow (D, C)$, where $M = (V, E, \psi)$ such that $V = \{x_1, x_2\}$, $E=\{(x_1, \#, x_2)\}$, $\psi(x_1)=A$, and $\psi(x_2)=B$, and $D=(V_D, E_D, \psi_D)$ such that $V_D = \{\ x_3, x_4, x_5\ \}$, $E_D=\{\ (x_3, \#, x_4), (x_4, \#, x_5), (x_5, \#, x_3)\ \}$, $\psi_D(x_3)=\#$, $\psi_D(x_4)=\#$, $\psi_D(x_5)=D$, and $C = \{\ c_1, c_2\ \}$ such that $c_1=(\#, \#, x_1, \#, x_5, in)$ and $c_2=(C, \#, x_2, \#, x_4, out)$.

Graph H in Figure 2 is generated by applying p to graph G.

Definition 5 The *language* of graph grammar $GG = (\Sigma, \Delta, \Gamma, \Omega, P, S)$ is $L(GG) = \{\ g \mid g$ is derived from S by GG , all node labels of g are terminal node labels, and all edge labels of g are final edge labels. $\}$

4 A Grid Graph Grammar

In this section, we construct a graph grammar for two-dimensional grid graphs and describe some graph grammar characteristics.

Definition 6 A *grid graph grammar* GG_G is defined as follows: $GG_G = (\Sigma_G, \Delta_G, \Gamma_G, \Omega_G, P_G, S_G)$, where $\Sigma_G = \{\ S, H, L, A, B, R, C, X, D, R', E, P_h, END, \#\ \}$, $\Delta_G = \{\ \#\ \}$, $\Gamma_G = \{\ \#\ \}$, $\Omega_G = \{\ \#\ \}$, P_G is defined in Figure 3, and $S_G = (\{v\}, \phi, \psi_{S_G})$ such that $\psi_{S_G}(v) = S$.

Here we explain the labels in P_G. Label S is the start node label. Label L is the leftmost node label of grid graphs. Label H is the label for making grid graphs that are not path graphs. Labels R and R' are the rightmost node labels of grid graphs. Labels A, B, P_h are for growing grid graphs to the left. Labels C and X are for turning right. Label END is the last nonterminal node label, and label # is the terminal node label.

Note that the unlabeled nodes and edges are labeled #.

We show an example of a derivation of GG_G by Figures 4, and 5 in the Appendix. This derivation is for the grid graph $T_{3,4}$. Figure 3 illustrates the productions of GG_G. In Figure 3, we denote the connection instructions of the productions using dotted lines.

Theorem 1. Graph T is a grid graph iff T is an element of $L(GG_G)$.

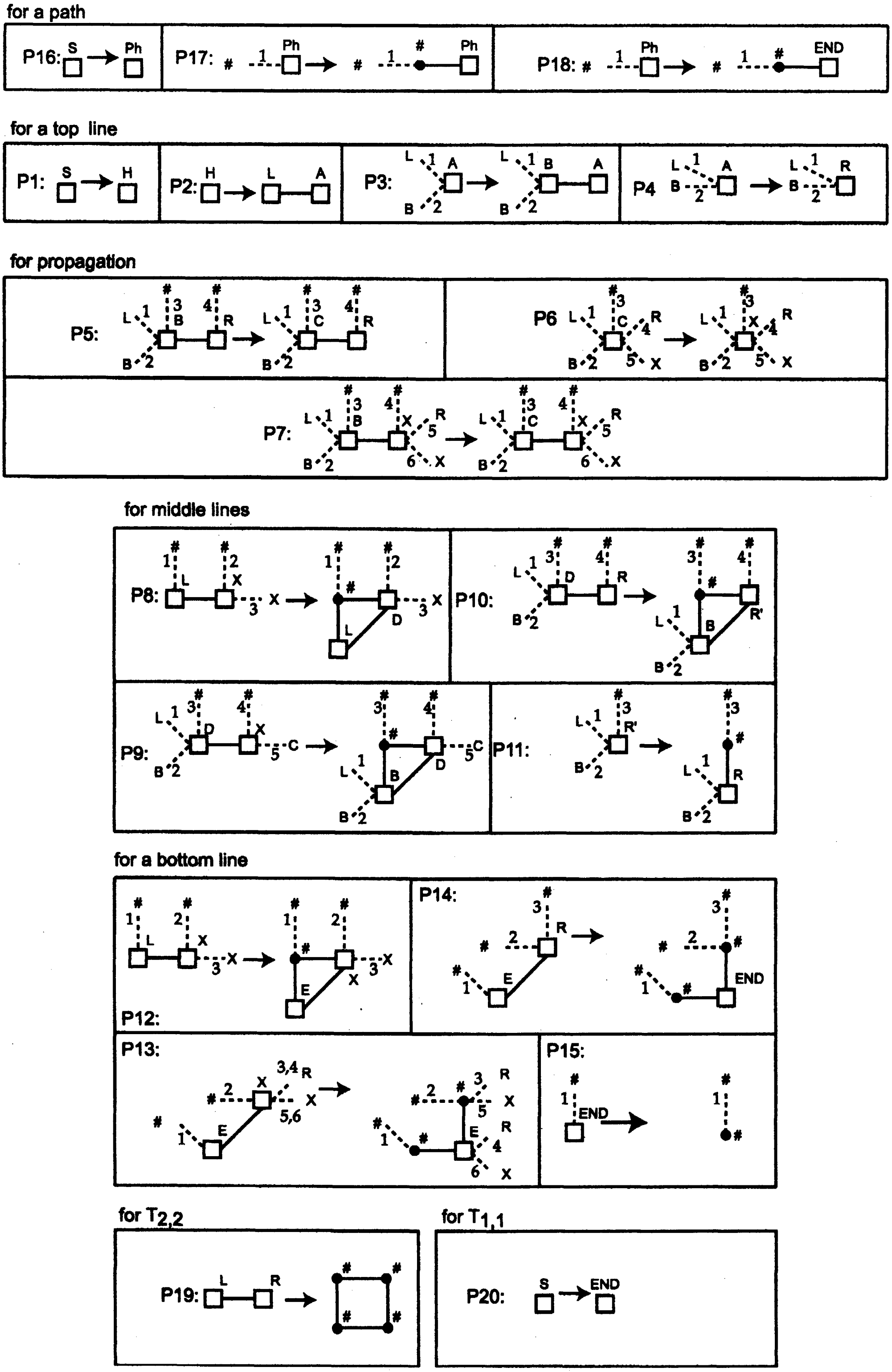

Figure 3: Productions of GG_G.

We can extend this graph grammar to graph grammars with multiple node labels.

Burosch and Laborde proposed a decision algorithm for two-dimensional grid graphs that runs in linear time for the number of nodes [7].

There are several applications of labeled grid graphs (e.g. symbol tables, bitmap image data etc.) that naturally relate labeled grid graphs. For example, the colors of pixels in bitmap images are represented by node labels.

Our graph grammar GG_G can be extended to several applications.

5 Conclusion

We constructed a graph grammar for two-dimensional grid graphs.

In the future, we will apply this grammar to grid graphs with multiple node labels such as financial statements. The results could possibly be applied to the parsing of grid graphs with multiple labels such as business document tables (e.g. [10, 11]) or symbol tables in program documents.

Acknowledgments

The authors would like to thank Professor Kimio Sugita of Tokai University and Associate Professor Koichi Yamazaki of Gunma University for their valuable discussions and helpful comments.

References

[1] T. Pavlidis, "Linear and Context-free graph grammars", J. ACM, vol. 19, no. 1, pp.11-22, 1972.

[2] T. Yaku, "The Constructibility of a Configuration in a Cellular Automaton", J. Comput. Syst. Sci., vol. 7, no. 5, pp.481-496, 1973.

[3] R. Franck, "A Class of Linearly Parsable Graph Grammars," Acta Infomatica, 10, pp.175-201, 1978.

[4] P. D. Vigna and C. Ghezzi, "Context-Free Graph Grammar", Inf. Control, 37, 207-233, 1978.

[5] J. Engelfriet, G. Leih, and G. Rozenberg, "Apex Graph Grammar", Lecture Notes in Computer Science, vvol. 291, pp.167-185, 1986.

[6] J. Engelfriet and G. Leih, "Linear Graph Grammars: Power of Complexity", Information and Computation, vol. 81, pp.88-121, 1989.

[7] G. Burosch and J.M. Laborde, "Characterization of grid graphs", Discrete Math. 87, pp.85-88, 1991.

[8] K. Yamazaki and T. Yaku, "A Pumping Lemma and the Structure of Derivation in the Boundary NLC Graph Language", Information Sciences 75, pp.81-97, 1993.

[9] G. Rozenberg (Ed.), Handbook of Graph Grammar and Computing by Graph Transformation, World Scientific Publishing, New Jersey, vol. 1, 1997.

[10] M. Shimizu, K. Tsuchida, and T. Yaku, "A Formalization of Business Documents by Graph Grammars", IPSJ SIG Technical Report, 2004-IS-88, 10, pp.66-73, 2004 (in Japanese.)

[11] T. Kirishima, T. Motohashi, T. Arita, K. Tsuchida, and T. Yaku, "Syntax for Tables", Proc. 21st IASTED Applied Informatics 2003, pp.1185-1190, 2003.

[12] T.Arita, K.Tomiyama, T. Yaku, Y. Miyadera, K. Sugita, and K. Tsuchida, "Syntactic Processing of Diagrams by Graph Grammars", the Proceedings of 16th IFIP World Computer Congress 2000 ICS2000, pp.145-151, 2000.

[13] T. Arita, K. Tsuchida, and T. Yaku, "Syntactic Characterization of the Two-Dimensional Grid Graphs", IEICE Transactions on Information and Systems, vol.E89-D, No.2, pp.771-778, 2006.

Appendix A: A Derivation Based on GG_G

This part shows the derivation process of the grid graph with three rows and four columns.

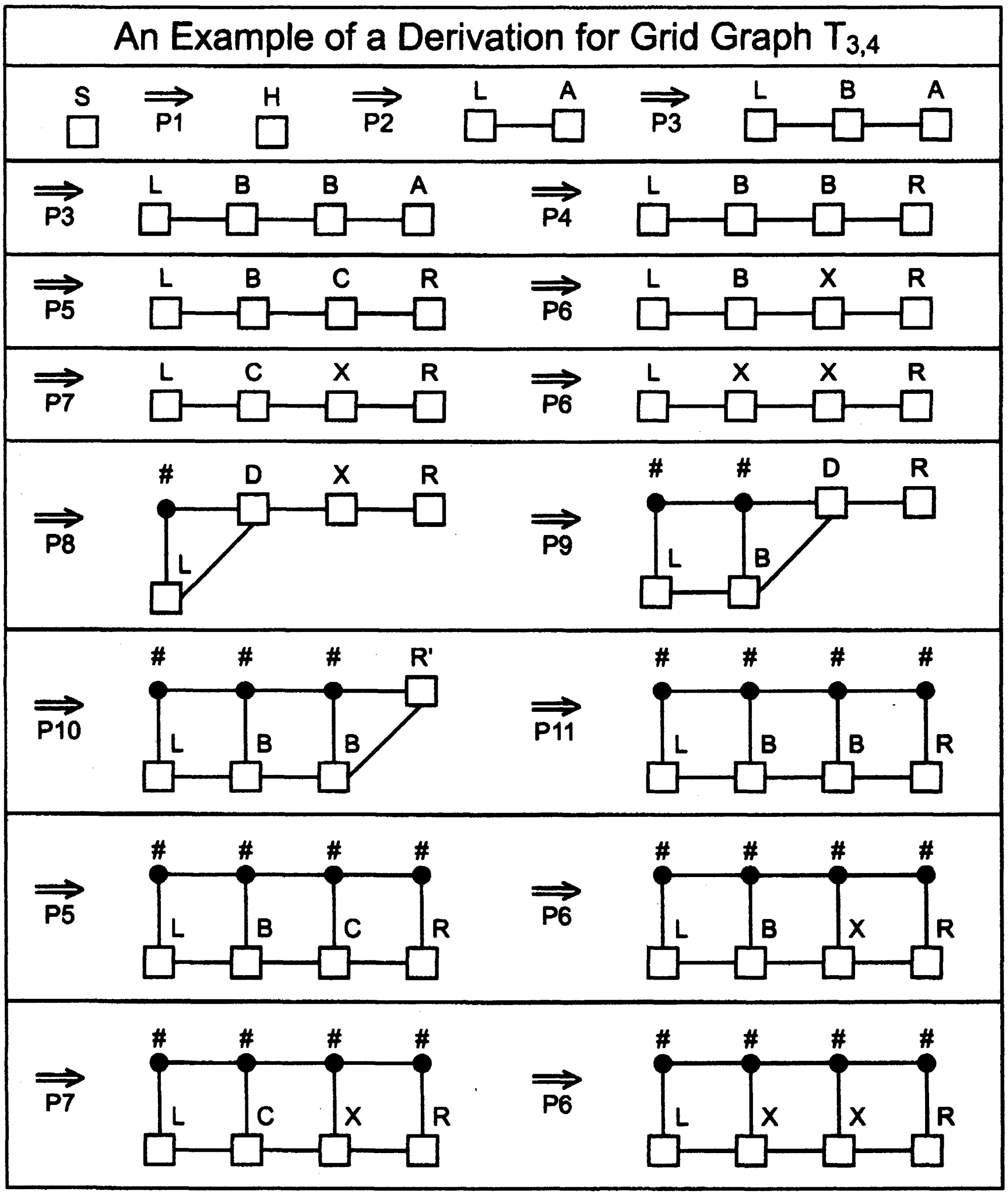

Figure 4: A derivation based on GG_G (1).

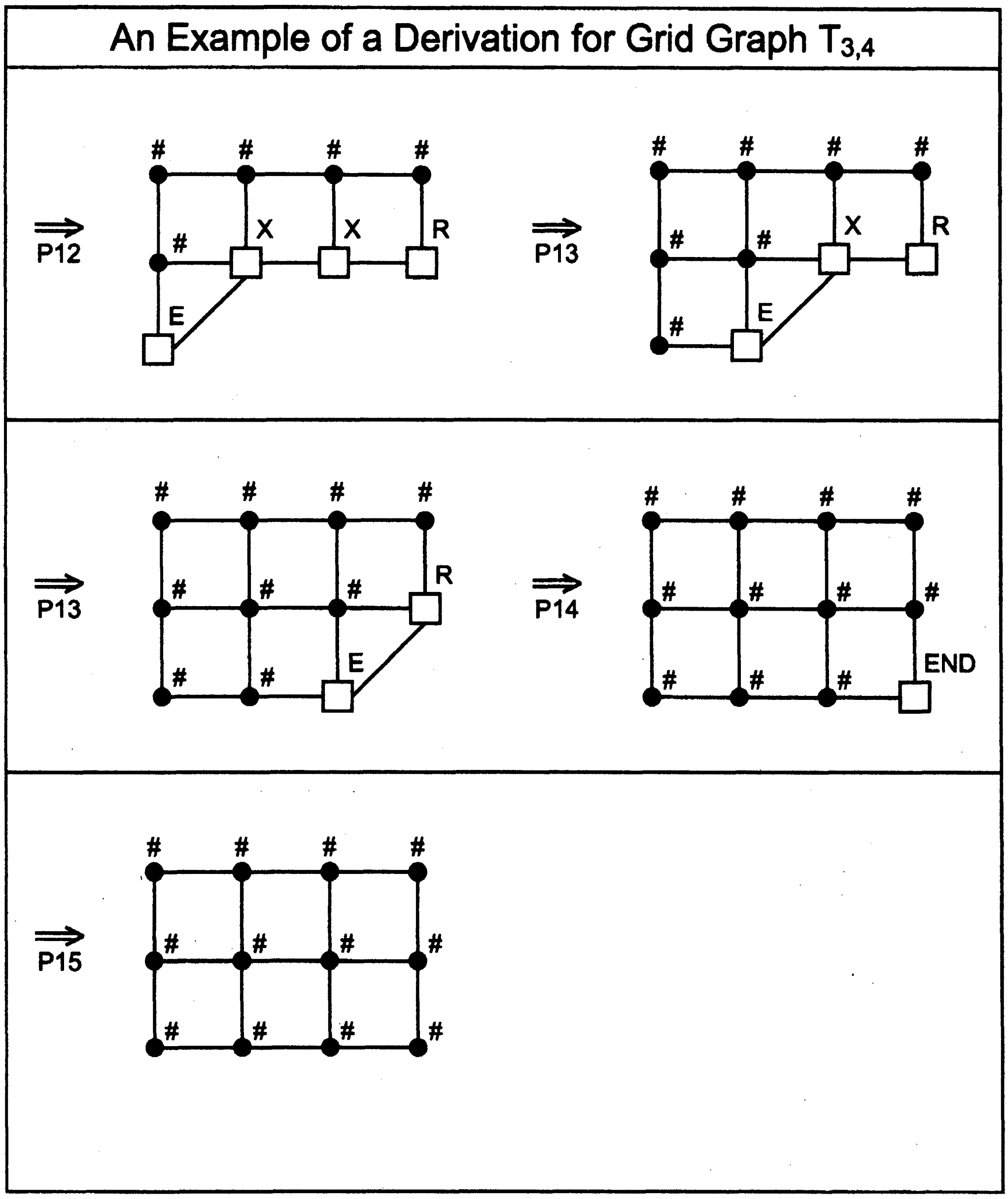

Figure 5: A derivation based on GG_G (2).