**TELKOMNIKA**, Vol. 11, No. 8, August 2013, pp. 4260~4266
e-ISSN: 2087-278X



# Construction of the Graphical Propositional Logic

**Yi Wang*[1], Han Ding[2]**
School of Mathematical and Computer Sciences, Hubei University of Arts and Science,
Xiang Yang, 441053, China, No.296 of Longzhong Road
*Corresponding author, e-mail: dhnats@163.com

***Abstract***

*Mathematical logic is the logical basis of the modern computer. It is important for the development of the electronic computer. With the development of computer technology, especially the development of computer visualization technology, the two-dimensional objects, such as graph and table, are more and more frequently as the computer processing object. But, the traditional one-dimensional character grammar already can not process. Then graph grammars that based on the two-dimensional arise at the historic moment. In order to provide the visual logic reasoning mathematical foundation, this paper introduces the basic theory of graph grammars. Through the propositional calculus and propositional logic reasoning two aspects of graphical description, we use graph grammars to construct propositional calculus and propositional reasoning.*

***Keywords****: proposition logic, reasoning theory, graphical*



## 1. Introduction

Computer language plays a crucial role for the development of the computer science. Formal language theory lays a theoretical basis for computer language. Because the Chomsky formal grammar proposed for the one-dimension string grammar, no matter from the processing object, or its own means of dealing with are based on one-dimension. But along with the rapid development of computer technology and application domain expansion, especially the rapid development of the visualization man-machine interface, the two-dimension figure and table used more widely and deeply, one-dimension string grammar already can not satisfy the need. Then, graph grammars based on processing for the figure and table arose at the historic moment. In the research of graph grammars, the two-dimensional objects represented as a figure with nodes and edges. Among them, nodes represent the objects; edges represent the relationship between nodes. Graph grammars are used to define and analyze this type of graph.

Graph grammars put the figure and table as the processing objects, define and analyze the objects. It abstracts the two-dimension object to the graph (directed or undirected graph) with the nodes and the edges, of which the nodes represent the object elements and the edges said the relationship among the objects. In addition, in the application background, it establishes a set of rules, using this set of rules to derive from the initial graph, and then get the languages of the graph grammar. Through the rules may also establish whether a graph belongs to a language produced from a given graph grammar.

Mathematical logic is the logical basis of the modern computer. It is important for the development of the electronic computer. In order to provide the visual logic reasoning mathematical foundation, this paper introduces the basic theory of graph grammars. Through the propositional calculus and propositional logic reasoning two aspects of graphical description, we use graph grammars to construct propositional calculus and propositional reasoning.

Based on the basic idea of graph grammars, we can graphical the proposition formulas and the reasoning formulas, and put the reasoning formulas as the graph grammar rules. With the derivation methods of the graph grammar, we can construct the graphical reasoning process for proposition logic. Section II briefly introduces the basic concepts and the research status of graph grammars. Section III to section V give the graphical methods for the propositional formulas, rules and the reasoning formulas. Section VI analysis some questions about using the rules. Section VII is the summary of the full paper, and gives the future research directions.

*Received March 4, 2013; Revised April 29, 2013; Accepted May 17, 2013*

## 2. Graph Grammars

Definition 1 A graph G = (N, E, l, s, t, L). Where, N is the set of the nodes; E is the set of the edges; l is the node label function; s: E → N and t: E → N are the starting node and the end node of the edges; L is the set of the labels for the nodes and the edges.

Definition 2 A production P = <L, R>, usually expressed as L: = R, where either of L and R is a graph, called the left-hand and the right-hand of a production.

Definition 3 A graph grammar GG is a triple (A, P, E), where A is the initial graph or called the original graph, P is a set of the productions, E is the embedding rules of the graph grammar.

The derivation process of a graph grammar is started from an initial graph, select the rules from the rule set, replace the subgraph of the initial graph. The subgraph is isomorphic with the left-graph of a rule; it is replaced by a graph, which is isomorphic with the right-hand of the rule, by using the embedding rules, and then gets a new graph. The set of the new graphs is called the language of the initial graph. The parse process is exactly the opposite, given a graph, select the rules from the rule set, replace the subgraph of the graph. The subgraph is isomorphic with the right-hand of a rule, by a graph, which is isomorphic with the left-hand of the rule, by using the embedding rules. If we can get the initial graph, we call the given graph as a language of the initial graph. A graph parse process can determine whether a given graph is a generated language from the initial graph. Strict distinction between the derivation and parse is not the time, they are all called as the graph transformation.

The so-called embedding rules that is a rule, when a sub-graph of the initial graph replaced, how the replacement part embedded into the initial graph and not generate the suspended edges. There was some success [1-5] in terms of the formal definition and the implementation of the graph grammars. In the application, they were also used in a variety fields [6-9]. Graph grammars can be divided into the context-free graph grammars and the context-sensitive graph grammars. In the early days of the graph grammars, the context-free graph grammars were researched mostly. The left-hand of a production of the context-free graph grammars is the only no terminal node. More common are: NLC (Node Label Controlled) [10], NCE (Neighbourhood Controlled Embedding) [11], HRG (Hyperedge Replacement Grammar) [12], RG (Relational Grammar) [13] and so on. In the Context-sensitive graph grammars, both the left-hand and the right-hand of a production define a set of about one-to-one graph elements, and complete the embedding procedure by the corresponding relations. The context-sensitive graph grammars has a stronger expression, and be suitable for the formal visual language description. More common are: LGG (Layered Graph Grammar) [14], RGG (Reserved Graph Grammar) and so on. EGG [15] is a context-sensitive graph grammar. Compared with the other Context-sensitive graph grammars, EGG production's structure is more simple, and it eliminate the context node, and may also reduce the number of production by no using the wildcard. The grammar is better understanding. The characteristics of EGG put the edges as the context elements, so that it abandons the semantic information of the nodes, and retains only the structural information. It is more convenient for production design.

## 3. Graphical Proposition Formulas

Based on the scope of studying for the propositional logic and the theory of graph grammars, we can visually express propositional logic by using the rules of graph grammars.

Two types of nodes are used to represent the propositions and the connectives. Which, ○ represents a proposition node, □ represent a connective node. The node labels represent the propositions and the connectives. Taking into account that the computing objects may be the constant values: TURE or FALSE (T or F), then in the label set of nodes, there are two special labels: T and F. The label set of connective nodes is {¬, ∧, ∨}; it said the type of operations. The label set of edges is a set of natural numbers. The edges With direction are divided into the following two types: first, the proposition node is the starting node, the connective node is the end node, that represents a proposition participates in the operation (referred to as: operator node); second, both the starting node and the end node are all the connective nodes, that represents the result of the prior connective node (referred to as: the result edge), while that represents the operation object of the second connective. A special case is: In a proposition formula, there is one and only edge, which only has the starting node without the end node, and represents the operation result of the proposition formula.

Such as: proposition formula P ∧¬Q ∨Q
Graphics into:

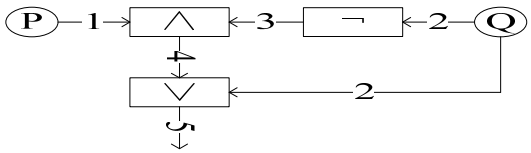



Figure 1. Graphical the Proposition Formula: P ∧¬Q ∨Q

It can be seen from Figure 1, the labeled 3 edge is the result of ¬operation with node of Q; the labeled 4 edge is the result of node of P carry out ∧operation with the labeled 3 edge; the labeled 4 carry out ∨ operation with node Q, then get a result, which is the result of the proposition formula and labeled 5.

## 4. Graphical Productions

A production consists of the left-hand and the right-hand, according to the definition and rules of EGG, either of the left-hand and the right-hand is a hanging-edge graph.

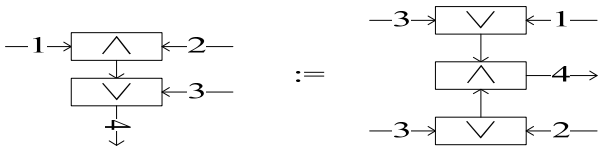


Figure 2. A Production

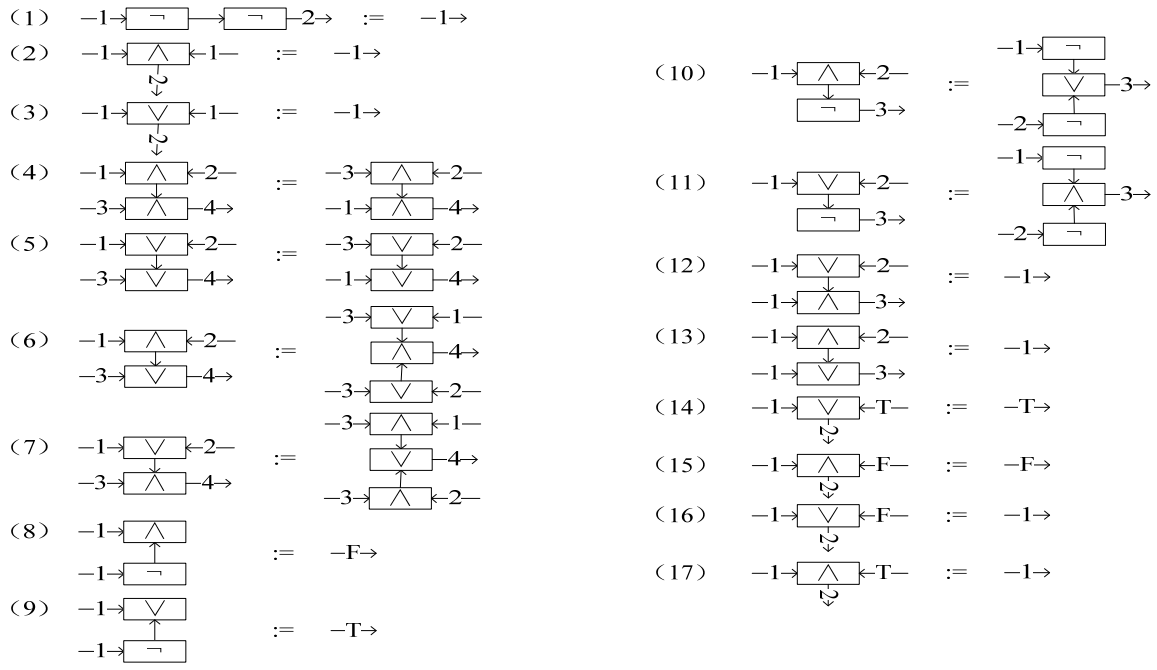



Figure 3. The Set of the Productions

The Figure 2 and Figure 3 are graphical production by using the distributive law of ∨ to ∧. The labels in the left-hand and the right-hand represent the corresponding relationship in the embedding process. When using a single production for the derivation or the parse, simply embed the according hand into the original graph by corresponding relation of the edges.

The productions of the proposition calculus can come from the equivalence formulas of the proposition logic. Summed up the equivalent formulas, we got the productions used to proposition calculus, in which we abandoned a number of equivalent formulas, these formulas can be derived from the other equivalent formulas available.

There are some common connectives: ¬, ∧, ∨, →, ↑, ↓, etc. But for the convenience, we use the full function set of the connectives to express. Here, we select {¬, ∧, ∨} as the full function set. If the other connectives appear in the proposition formula, we can convert them to contain only {¬, ∧, ∨}.

Graphical the conversion formulas are:

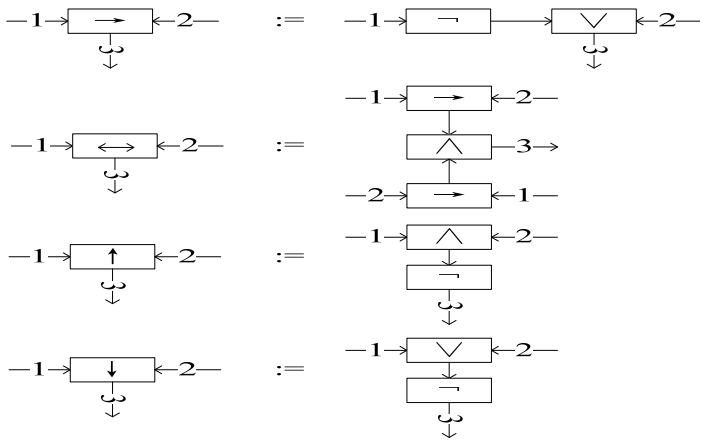



Figure 4. Graphic Connectives Conversion Formulas

In the process that the equivalent formulas convert to the productions, there are two special nodes: TRUE or FALSE (T or F), the node label can be expressed as T or F. The label of the outgoing edges from the special nodes can be expressed as T or F. There is also a type of special edges $\rightarrow$ , which have not the starting point and the end point. The embedding method of this type of nodes is that the starting point is determined by the correspondence relationship between the edge label, and the end point is determined by the result edge.

## 5. Graphical the Reasoning Rules

A reasoning rule consists of the left-hand and the right-hand. The left-hand may be a connected graph, and also be an unconnected graph composed with multiple connected subgraphs. If it is an unconnected graph, each of the connected subgraphs is a premise. The right-hand is a connected graph, it shows the reasoning result.

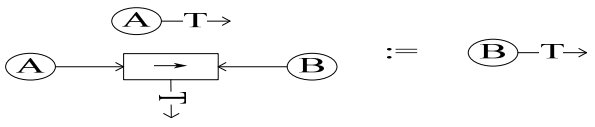
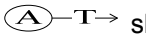



Figure 5. A Reasoning Rule

In Figure 5,  shows the proposition formula of A is true. The rule shows that it can derive the result of B from the premises of A and A → B. i.e. if the proposition formulas of A and A → B are true, the proposition formula of B is true.

The graphical form of the common reasoning rules as shown below:

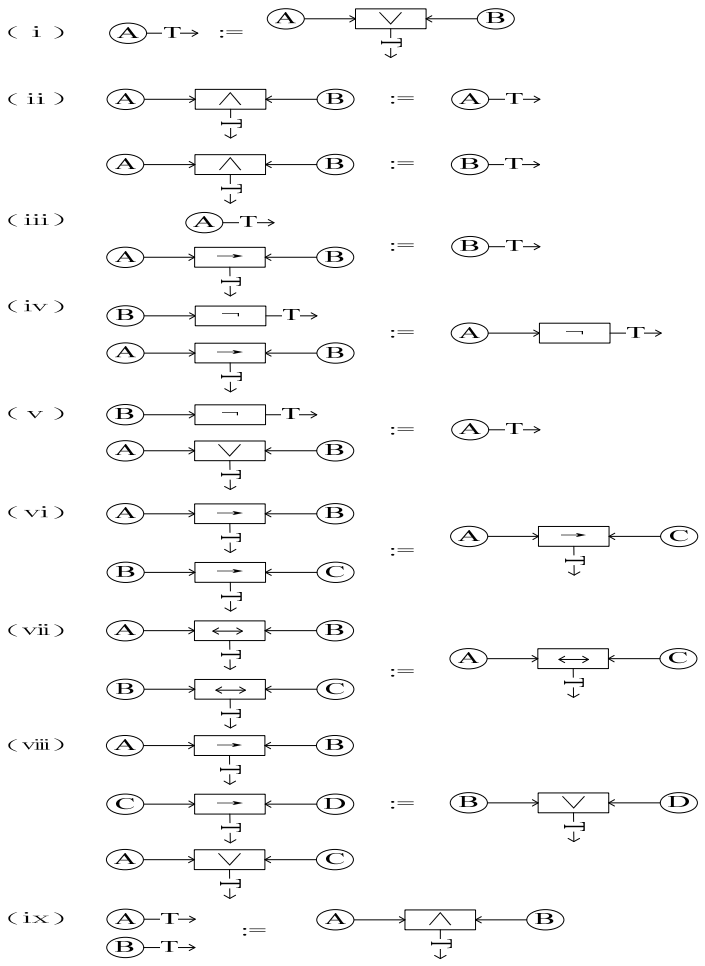



Figure 6. The Graphical form of the Common Reasoning Rules

By using the reasoning rules for many times, the process that derives the conclusion from the premises is called the reasoning construction.

## 6. Example and Analysis

Here is an instance of constructing a reasoning process:
For example:
A policeman examines a Cornhill robbery; the known facts are as follows:
(1)A or B stole the recorder;
(2)If A stole the recorder, the crime cannot occur before midnight;
(3)If B testimony is correct, the light in the room was not extinguished in midnight;
(4)If B testimony is not correct, the crime occurred before midnight;
(5)The light in the room was extinguished in midnight.
Based on the known facts above to judge who stole the recorder.
Firstly, using the symbols to represent the known facts:
P:A stole the recorder;
q:B stole the recorder;
r: The crime occurred before midnight;
s:B testimony is correct;
t:The light in the room was not extinguished in midnight.
So, the premises are shown as fellows: $p \vee q$, $p \rightarrow \neg r$, $s \rightarrow t$, $\neg s \rightarrow r$, $\neg t$
Graphical the premises:

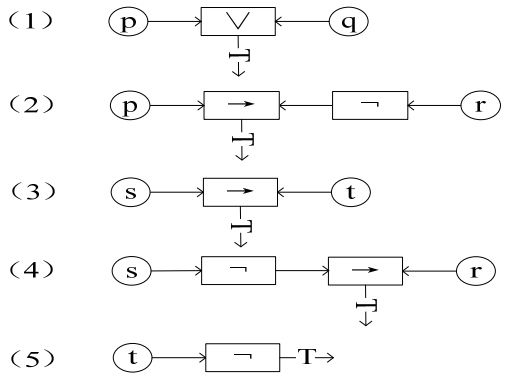



Figure 7. Graphical the Premises

Derived as follows:

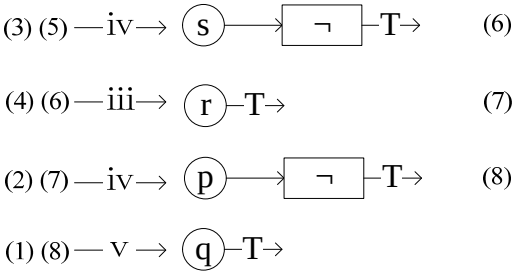



Figure 8. The Reasoning Process

We can see from the derivation that the proposition of q is correct. So, we can judge that B stole the recorder.

When using an inference rule, the match between the left-hand of the rule and the premises, only needs in the structure to carry on the isomorphism judgment, and the labels of the proposition nodes only needs to establish the corresponding relationship. By using the rules to obtain the conclusion, which according to this kind of the corresponding relationship to determine the labels of the nodes in the conclusion.

In the inference process, each proposition formula or is the known premise, or is the conclusion, which obtained by using the inference rules upon some certain premises. In any step of the inference, a premise can be introduced. In any step of the inference, the conclusion which proved all may be taken as the premise in the following inference process. In any step of the inference, any a sub-proposition formula in the proposition formula all may be replaced with its equivalent proposition formula.

## 7. Conclusion

This paper uses the rule derivation method of graph grammars to graphical the reasoning rules as the productions, and constructs the graphical inference process upon the premises. This paper summarizes and analyzes the usage of the rules in the inference process. In the future, we will construct a more perfect mathematical logic system, which based on the graph grammars, to build the foundation for constructing the automatic graphical reasoning process for proposition logic.

## References

[1] Rozenberg G. Handbook of Graph Grammars and Computing by Graph ransformation. World scientific Publishing Singapore. 1997; 11.

[2] Ehrig H, Engels G, Kreowski H2 J, et al. Handbook of Graph Grammars and Computing by Graph Transformation: Applications, languages and Tools. Singapore: World scientific Publishing. 1999; 2.
[3] Ehrig H, Kreowski H2 J, Montanari U, et al. Handbook of Graph Grammars and Computing by Graph Transformations: Concurrency, parallelism, and Dist ribution. Singapore: World scientific Publishing. 1999; 3.
[4] Drewes F, Hoffmann B, J anssens D, et al. Adaptive Star Grammar. ICGT. 2006: 77-91.
[5] Lara J, Bardohl R, Ehrig H, et al. Attributed Graph Transformation with Node Type Inheritance. *Theoretical Computer Science*. 2007; 376(3): 1392163.
[6] Pfaltz J. Web Grammars and Picture Description. *Computer Graphics and Image Processing*. 1972; 1(1): 193-220.
[7] Bunke H. Attributed Programmed Graph Grammars and Their Application to Schematic Diagram Interpretation. *IEEE Pattern Analysis and Machine Intelligence*. 1982; 4(6): 574-582.
[8] Fahmy H, Blostein D. A Graph Grammar Programming Style for Recognition of Music Notation. Machine Vision and Applications, to appear 1992 Preliminary result. Proc. First International Conference on Document Analysis & Recognition. St. Malo, France. 1991: 70-78.
[9] Dolado J, Torrealdea F. Formal Manipulation of Forrester Diagrams by Graph Grammars. *IEEE System, Man and Cybernetics.* 1988; 18(6): 981-996.
[10] Rozenberg G, Welzl E. Boundary NLC Graph Grammar Basic Definition, Normal Forms, and Complexity. *Information and Control*. 69: 136-167.
[11] J anssens D, Rozenberg G. Graph Grammars with Neighbourhood controlled Embedding. *Theoretical Computer Science*. 1982; (21): 55-74.
[12] Drewes F, Kreowski J, Habel A. Hyperedge Replacement Graph grammars. In[ 2 ]: 95-156.
[13] Wit tenburg K. Earley2Style Parsing for Relational Grammars. Proc. 8t h IEEE Workgroup on Visual Languages. Seattle, WA, Los Alamitos, CA. IEEE Computer Society Press. 192-199.
[14] Rekers J, Schü rr A. Defining and Parsing Visual Languages with Layered Graph Grammars. *Journal of Visual Languages and Computing*. 1997; 8(1): 27-55.
[15] ZENG Xiao-Qin, HAN Xiu-Qing, ZOU Yang. An Edge-Based Context-Sensitive Graph Grammar Formalism. *Journal of Software*. 2008; 19(8): 1893-1901.