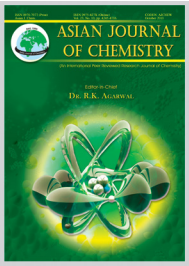
Asian Journal of Chemistry; Vol. 24, No. 11 (2012), 5332-5336

ASIAN JOURNAL OF CHEMISTRY

www.asianjournalofchemistry.co.in



Construction of the Isomers for Organic Compounds Based on Graph Grammars

YI WANG[1,2,*], XIAO QIN ZENG[1], ZHAN SHI[1], NING ZHU[1] and YUN ZHU[1]

[1]Institute of Intelligence Science and Technology, Hohai University, Nanjing 210098, P.R. China
[2]School of Mathematical and Computer Sciences, XiangFan University, Xiang Yang 441053, P.R. China

*Corresponding author: E-mail: yiprince@126.com

(*Received*: 21 November 2011; *Accepted*: 16 June 2012) AJC-11629

There is a type of isomerism in organic chemistry, in which the number of isomers is exponentially increasing when the number of carbon atoms becomes large. This paper proposes a method by employing the theory of graph grammars with designing some production rules to derive the isomers of the organic compounds that are only consist of carbon and hydrogen atoms. In addition, the method also optimizes and extends the rules to construct a carbon framework for the isomers of the organic compounds that are consist of not only carbons and hydrogen atoms. When the carbon framework is determined, it is easy to construct the isomers by joining other atoms to the framework. Finally, the paper introduces the design of related softwares, which can construct the isomers automatically from the basic structure of the given organic compounds and record the parsing process.

Key Words: Organic compounds, Isomer, Graph grammar, Software design.

INTRODUCTION

There is a type of phenomenon called isomerism in organic chemistry. The organic compounds have the same molecules compositions, but the atoms arrangement is different, that is, the molecular structure of organic compounds is different. These organic compounds, which have the same molecular formulas, but have the different structures, are called isomers. In organic chemistry, the isomerism is very popular and it is one of the main reasons that the number of various organic compounds is large. The finding of the isomerism promotes the establishment for the organic chemical structure theory.

However, given a molecular structure formula, how many isomers are there? What is their structure? Between isomers, can they be conversed each other? If they can, how to do it? So far, these problems have failed to effectively resolve. Even, the number of the isomers was not sure yet. Although, some mathematicians and chemist calculate the number of the isomers through derivation and constructing a mathematical formula and some verify the correctness of the part through the manual, or find out the writing rules through the structure law, they can not get all specific isomers. It is time-consuming by the manual, even can not finish. With the number of isomers for the methane with series (C_nH_{2n+2}) as an example, when there are 20 carbon atoms in a organic compounds, the number of it's isomers is 366319 and when there are 40 carbon atoms in a organic compounds, the number of it's isomers is 62491178805831.

Because the computer has the fast operation ability and the massive storage capacity, we can consider it as a tool to derive the isomers for organic compounds. The traditional computer program is based on the one-dimensional linear string, but the structure of the compound is a kind of three-dimensional structure. Obviously, this one-dimensional approach already can not satisfy the needs of the three-dimensional structure to deal with this. According to this shortage of the one-dimensional grammar, Pfaltz and Rosenfeld put forward the concept of graph grammar. Graph grammar put the object spread to the two-dimensional graphics from the one-dimensional string. Although the organic compound is a kind of the three-dimensional structure, it can be expressed by using the two-dimensional formula according to the Kekule formula. Based on the two-dimensional formula, we can use the principle of graph grammar to derive the isomers for organic compounds and record the derivation process. In theory, we can derive out all the isomers for an organic compound by using this method and can derive automatically and rapidly by using the fast operation ability and the massive storage capacity of the computer.

Because of the importance of carbon framework for constructing the isomers, this paper starts from constructing the isomers for organic compounds, which is only consist of carbon, hydrogen atoms. For the organic compound, which is

not only consists of carbon, hydrogen, we can determine its carbon framework by using this method.

Based on the theory of the graph grammars, this paper focuses on graphical representation the organic compounds and the rules. We only consider the structural transformation and not consider the specific chemical reactions. (i) introduces the basic theory of the graph grammars; (ii) introduces the method that makes the molecular structures and the rules graphical; (iii) makes some rules and analyzes them briefly; (iv) introduces the optimization and promotion for the rules; (v) introduces the software design; (vi) summarizes the work and looks forward.

Graph grammars

Definition 1: The 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 edges.

Definition 2: The 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 embedded rules of the graph grammar.

The derivation process of a graph grammar is started from an initial graph, select the rules of the rule set, replace the subgraph of the initial graph. The subgraph is isomorphic with the left-hand 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 in the rule set, replace the subgraph of the graph. The subgraph is isomorphic with the right-hand of a rule, it is replaced 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 by the initial graph. Strict distinction between the derivation and the parse is not the time. They are all called as the graph transformation.

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 non-terminal node. More common are: Node Label Controlled (NLC)[10], Neighbourhood Controlled Embedding (NCE)[11], Hyperedge Replacement Grammar (HRG)[12], Relational Grammar (RG)[13] *etc*. 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 grammar has a stronger expression and is suitable for the formal visual language description. More common are: Layered Graph Grammar (LGG)[14], Reserved Graph Grammar (RGG) *etc*. 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 better understands. 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.

Graphic molecular structure and the rules

Organic compounds mainly consist of the elements of C, H, O and N. In the graphic an organic compound molecular structure, ◯ said the atom of an element, ◯ have two attributes: number and label. Number is the sequence number of the atom in a molecular; Label shows the type of an atom, such as C, H and O. For convenience, it omits H atoms here. Edges in the graph represent the atomic bonds between atoms and they are undirected edges. Edges have an attribute: label to establish the corresponding relation between the initial graph and the left-graph in a rule in a derivation.

For example: C_5H_{12} alkane can be expressed as Fig. 1.

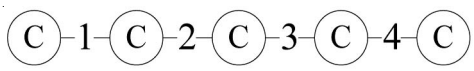


Fig. 1. The graphical C_5H_{12}

According to the principle of EGG, both sides of a graphical rule are a suspended-edge graph. The labels of the suspended-edges are used to establish the corresponding relation with the labels of the edges in the initial graph, such that, it is convenient to realize the embedding.

For example: C_4H_{10} alkane can be expressed as:

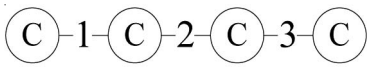


Fig. 2. The graphical C_4H_{10}

A graphical rule:

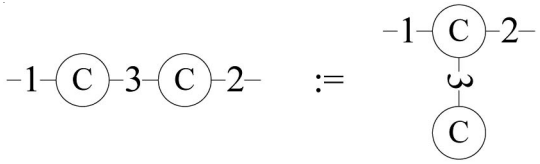


Fig. 3. A graphical rule

Derivation process: Finding out the subgraph of the initial graph which is isomorphic with the left-graph of a rule (in Fig. 3 only one this type subgraph), then replace the subgraph with the right-graph of the rule and complete the embedding according the corresponding relation between edges in the left-graph and the right-graph. We can get the isomer for C_4H_{10}.

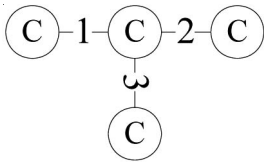


Fig. 4. The isomer for C_4H_{10}

Rules making is a complicated process. The made rules must be satisfied the completeness, that is, the made rules must be satisfied the all structure transformation. But, so may have a higher redundancy. For example, starting from a basic structure, by using the different rules or the different sequence of the rules, we can derive the same isomer. So, that is to the derivation, it has redundancy, but for the other side, it can provide more derivation paths and provide auxiliary for the choice of the synthesis paths. In the next chapter, starting with the derivation of the isomers of the alkane hydrocarbons, we will analyze the rule making the use of the rules and the optimization of the rules in detail.

Making rules

Because the organic compounds mainly consist of C and H atoms, we first consider the derivation rules for the alkane hydrocarbons contain only C and H atoms and then extend.

Given an alkane hydrocarbon, we can draw its straight chain structure graph easily. Making this straight chain structure graph as the initial graph, according to the rules, we can derive its all the isomers. Through the summary to the alkane hydrocarbons structure, the following derivation rules are concluded:

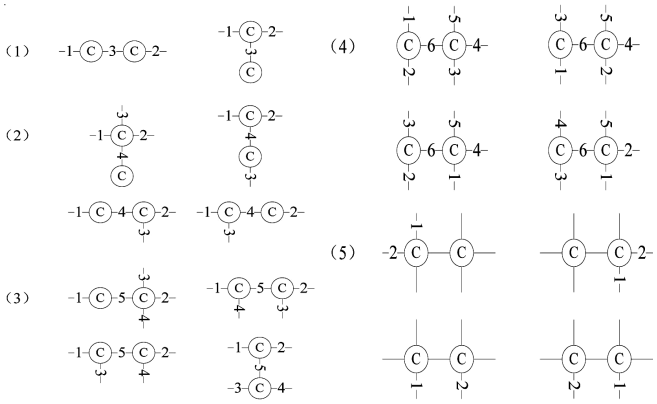


Fig. 5. Derivation rules of the isomers

Annotation:

(1) For the suspended-edges graph which has only one node, it has only one form on the structure, so it has no structural transformation.

(2) Structures in a group can be transformed each other, that is, in a group, any two suspended-edge graphs can be as the left-graph or the right-graph of a rule.

(3) The rules of (2), (3), (4), can be got from the prior rules through adding a suspended-edge.

(4) When we use the rules, we can choose the rules according to the quantity of the edges on two adjacent nodes, then determine the corresponding relation between the edges and then embed.

Theorem: The rules above are complete.

Proof: Carbon has four valency, so there are four atoms connected with it. In graph, they are expressed as four sides. An atom C has no transformation on the structure, so we do not consider this case and the structures which consist of three or above atoms may be composed of one atom and two atoms. So, in this paper, we consider only the structure transformation for two atoms. Because atom H complements to go into the structure easily, we omit the H atoms as the structural transformation. For the suspended-edges graph which consists of two atoms of C, the domain of the edges[1,7]. When it is 1 or 2, the compound is ethane or propane, but they have no isomers, so we do not consider them. Sum up above, we consider the domain as mentioned by Ehrig *et al.*[3] and Bunke[7], they correspond with the rules of (1) to (5) respectively. The structures in a group can be transformed each other.

Here putting the derivation process of the isomers for C_5H_{12} as an example, we introduce the usage of the rules simply.

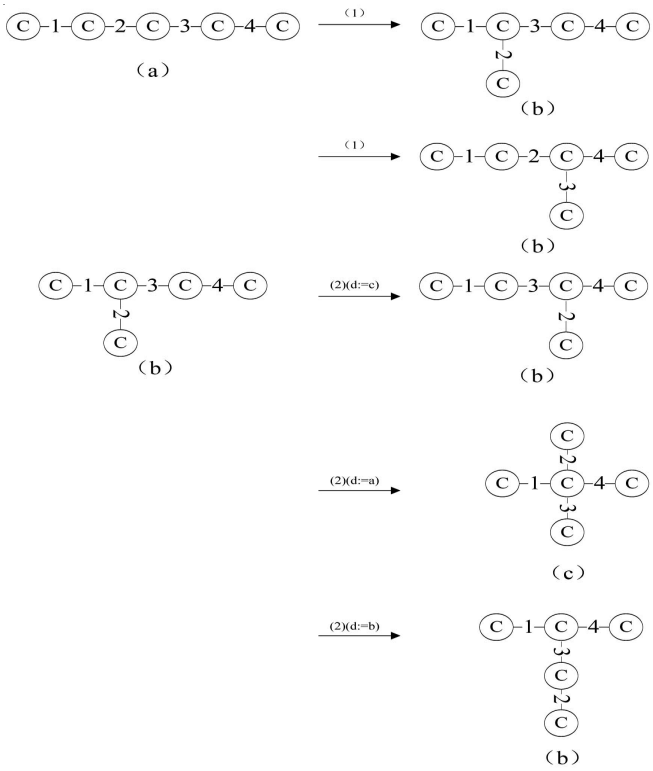


Fig. 6. Derivation process of the isomers for C_5H_{12}

From the derivation process, we can see out: (1) It can derivate all the isomers through the rules. (2) For the same isomer, there are not only one derivation path. This may be redundant, but it provides an idea to design the organic compounds synthesis path. These works will be introduced in the other paper.

Optimization and promotion for the rules

Optimization for the rules: While using the embedding rules of EGG, it allows the suspended edges expressed the context in the redex connect with the residual graph and do not allows the other edges connect with the residual graph. On the one hand, that solves the suspended edge problems in

the process of the embedding, but on the other hand, that makes the application limited. If we make the embedding conditions relax appropriately and add the processing method for the suspended edges, then the application scope will be enlarge. We will optimize the rules to make the quantity reduce and reduce the time-complexity for finding out the redex repeatedly. The basic idea is: Making the redex found out in the initial graph as a whole, the edges (except for the suspended edges been the context) that connected the residual graph with the redex are considered as connecting the whole. From the whole point of view, after the redex replaced, the edges will still link to the whole, but to consider what the node connected. Single from the structure, it can bring this kind of edge connect to any a node in the whole, each connection way shows a kind of structure transformation. So that, the prior derivation rules can be induced as one set. This set of rule is:

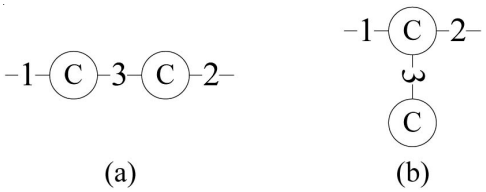


Fig. 7. The set of the optimized rules

The two suspended edge graph in the set of the rule may be the left-graph or the right-graph of the rule respectively.

The other rules can be derived out from the set of the rule, for example, one of the rules (2) in the prior section can be derived out:

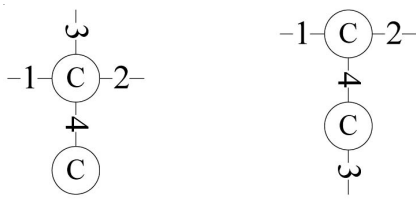


Fig. 8. Example of using the optimized rules

Promotion and expansion for the rules

This set of rules not only used to derive the isomer for the alkane hydrocarbons, but also apply to construct the carbon frame in the process of deriving the isomers for the other organic compounds. Such as: olefins, alkynes. Derivation method is introduced below.

In the process of deriving the isomers for the olefins and the alkynes, we put the double bonds or the triple bonds between the atoms as the edges, these edges link to the same node. While derivation, the node in the residual graph is not considered; while embedding, these edges all link to the node.

For example: Constructing the isomers for propylene:

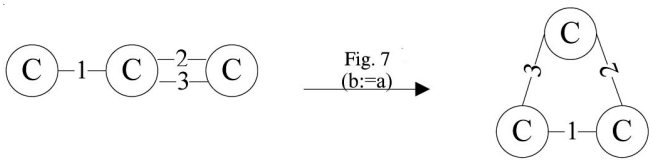


Fig. 9. Constructing the isomers for propylene

We can see out from the prior sections that the rules in Fig. 7 are effective for the organic compounds only consist of the C and H atoms. For the organic compounds not only consist of the C, H atoms, the rules can establish the carbon frame of the isomers. For example, for haloalkane, the carbon frame can be established through the method firstly, then get its isomers by instead of the atoms of H by the halogen atoms.

For example: Constructing the isomers for one of the atoms of H in pentane instead of a halogen atom.

Firstly, constructing the isomers for pentane according to the method in the paper, that shown in Fig. 6.

Secondly, replace an atom of H by a halogen atom to get the isomers, that shown in Fig. 10.

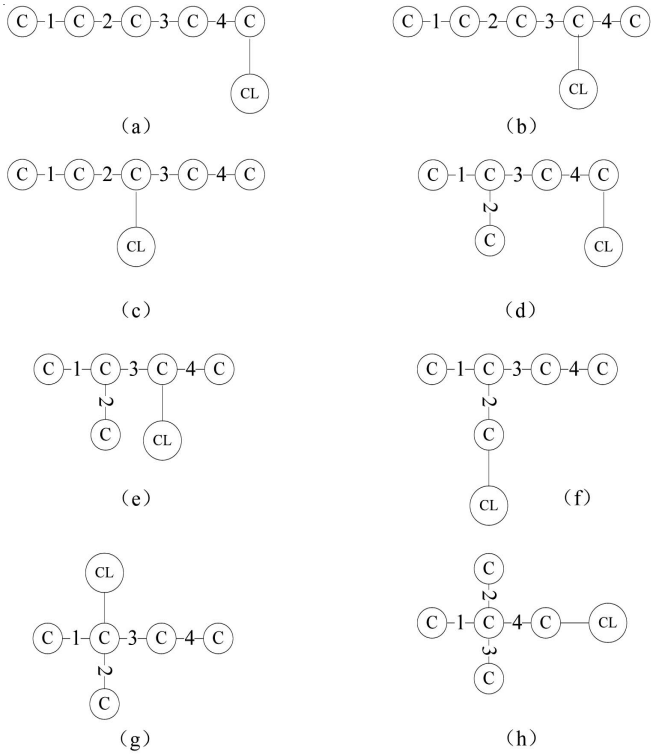


Fig. 10. The isomers of a haloalkane

From this example, it is observed that this method is effective for constructing the isomers for the organic compounds.

Software design

According to the prior derivation rules and methods, for improving the efficiency, we need to design a set of software to realize the derivation process. We will introduce the software design from the following several aspects.

Design a dialog to draw graphs and rules, in the dialog, we can draw the nodes and the edges by using mouse and set the attributes of the nodes and the edges. The drown graphs and rules need to stored in the external storage.

External storage design for graphs

(1) Every graph is stored in the disk with the XML document.

Mark of Graph: Every XML document of a graph is composed with <graph> and </graph>.

Attributes of Graph: The mark of the graph has three attributes: graphid, nodenum and edgenum, respectively

represents the label of the graph, the quantity of the nodes and the edges in the graph.

For example:

<graph graphid=1, nodenum=4, edgenum=3>

...

</graph>

It represents a graph, the label is 1 and the graph has four nodes and three edges.

(2) Mark of a node

Every node is composed with <node> and </node>:

<node nodeid=?>

</node>

The attribute of nodeid represents the number of the node.

(3) Mark of an edge

Edges have three attributes: edgeid, from and to, respectively represents the label of the edge, the numbers of the nodes of the edge.

Every edge is composed with <edge>, </edge>:

<edge edgeid=2, from=4, to=3>

</edge>

It represents an edge, the number is 2 and the edge connects the node 4 and the node 3.

External storage design for rules

(1) Every rule is stored in the disk with the XML document.

Mark of productions: Every XML document of a production is composed with <productions> and </productions>.

Attributes of productions: The mark of productions has two attributes: productionsid and productionsnum, they represent the number and the quantity of the production group separately.

(2) Mark of a production

Every production is composed with <production> and </production>. The mark has an attribute: productionid, it represents the number of the production.

(3) Mark of the left-graph

Every the left of the production is represented with <left> and </left>. At the same time, because the left of the production is a graph, the graph can be represented through the corresponding mark in Graph.xml. It is shown in fellows:

<left>

<graph graphid =231, nodenum =5, edgenum =10>

...

</graph>

</left>

(4) Mark of the left-graph

Every the right of the production is represented with <right> and </right>. At the same time, because the right of the production is also a graph, the graph can be represented through the corresponding mark in Graph.xml. It is shown in follows:

<right>

<graph graphid=231, nodenum=5, edgenum=10>

...

</graph>

</right>

Construction process

This software can realize to construct the isomers automatically:

(1) The drawing function: In the corresponding interface we can draw the graphs and the productions and set their attributes. (2) The drawn graphs and productions are stored in the disk with the XML document. The XML format can be use to access the graphs and the productions conveniently. (3) Starting from the initial graph, choosing the rule, looking for the subgraph of the initial graph, which is isomorphic to the left-graph of the rule and embedding the right-graph to the residual graph. Through the method, we can get an isomer of the initial graph and store it into the external disk. (4) Puts the gotten isomer as the initial graph, repeat the step (3), until can not get the new isomers.

Summary and future work

In this paper, we construct the isomers for the organic compounds by designing software based on graph grammars. This method can construct the isomers for the organic compounds which only consist of the atoms of C and H and also the organic compounds which consist of the atoms other than C and H. This method with the aid of computers can greatly reduce the difficulty and the time-complexity of handwork. Through optimizing the rules, the quantity of the rules is reduced, and this makes the subgraph isomorphism algorithm, which is N-P complete, be high efficiency.This method is useful for finding and recording the transformation paths for the isomers, and significant for choosing the synthesis path. However, the applied scope of the method is somewhat narrow. Our next work is to construct the synthesis software by using the graph grammars based on the existing computer-aided organic synthesis database. It is obvious that using the two-dimensional graphs is easier and more efficient for representing, storing and recognizing the chemical reactions than the one-dimensional strings.

ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China under grant No.61170089.

REFERENCES

1. H. Ehrig, H.J. Kreowski, U. Montanari and G. Rozenberg, Handbook of Graph Grammars and Computing by Graph Transformations: Concurrency, Parallelism and Distribution, Singapore: World Scientific Publishing, p. 3 (1999).
2. G. Rozenberg, Handbook of Graph Grammars and Computing by Graph Transformation, World Scientific Publishing: Singapore, Vol. 1 (1997).
3. H. Ehrig, G. Engels, H.J. Kreowski and G. Rozenberg, Handbook of Graph Grammars and Computing by Graph Transformation: Applications, Languages and Tools, World Scientific Publishing: Singapore, p. 2 (1999).
4. F. Drewes, B. Hoffmann, D. Janssens and M. Minas, *Theoret. Computer Sci.*, **411**, 34 (2010).
5. J. de Lara, R. Bardohl, H. Ehrig, K. Ehrig, U. Prange and G. Taentzer, *Theoret. Computer Sci.*, **376**, 139 (2007).
6. J. Pfaltz, *Computer Graph. Image Process.*, **1**, 193 (1972).
7. H. Bunke, *IEEE Pattern Anal. Machine Intellig.*, **4**, 574 (1982).
8. H. Fahmy and D. Blostein, *Machine Vision and Applications*, **6**, 83 (1993).
9. J. Dolado and F. Torrealdea, *IEEE System, Man Cybernetics*, **18**, 981 (1988).
10. G. Rozenberg and E. Welzl, *Inform. Control*, **69**, 136 (1986).
11. D. Janssens and G. Rozenberg, *Theoret. Computer Sci.*, **21**, 55 (1982).
12. F. Drewes, J. Kreowski and A. Habel, In ed., G. Rozenberg, Hyperedge Replacement Graph Grammars. Handbook of Graph Grammars and Computing by Graph Transformation, World Scientific Publishing: Singapore, Vol. 1, Ch. 2, pp. 95-156 (1997).
13. K. Wittenburg, Earley-style Parsing for Relational Grammars, Proceedings of IEEE Workshop on Visual Languages, September 15-18, Seattle, USA (1992).
14. J. Rekers and A. Schurr, *J. Visual Lang. Comput.*, **8**, 27 (1997).
15. X.-Q. Zeng, X.-Q. Han and Y. Zou, *J. Software*, **19**, 1893.