Journal of Computer and Mathematical Sciences, *Vol.10(2),270-278 February 2019* *ISSN 0976-5727 (Print)*
(An International Research Journal), *www.compmath-journal.org* *ISSN 2319-8133 (Online)*

Learning Fullerene Structures by Graph Grammars

Thanga Murugeshwari. V[1*] and Emerald Princess Sheela J.D[2]

[1]Research Scholar ,
Department of Mathematics, Queen Mary's College,
Chennai-4, Tamil Nadu, INDIA.
[2]Assistant Professor,
Department of Mathematics, Queen Mary's College,
Chennai-4, Tamil Nadu, INDIA.
email:[1*] thangammathe05@gmail.com., [2]emeraldsolomon96@gmail.com.

(Received on: February 8, 2019)

ABSTRACT

In this paper, some infinite classes of fullerene graphs are generated using hyperedge replacement graph grammars and a learning algorithm is discussed to learn some infinite classes of fullerene graphs. This paper is an impact of Jeltsch and Kreowski work on Grammatical Inference based on Hyperedge replacement ,to study some infinite classes of fullerene graphs.

AMS (2010): 68Q42, 68R10, 97R40.

Keywords: Graph Grammars, Hyperedge replacement, Fullerene.

1. INTRODUCTION

Graphs are used in various fields of science especially in computer mathematics, mathematical chemistry, and artificial intelligence. Graph grammars consists of rewriting rules that are called as production rules which replaces a part of graph with some other graph structure. Generation and manipulations of graphs are used by graph grammars.[1,5].

Fullerenes are made up of carbon atoms. Fullerenes graphs are graphs with vertex set is set of atoms and edge set is the set of bonds between the atoms. polyhedral molecules made entirely of carbon atoms. Researchers around the world are exploring applications of fullerene. In 1996 the Nobel prize in Chemistry was awarded for the discovery of fullerene. Fullerenes find wide application in different fields of science since their discovery in 1985. There are many applications of practical importance of fullerene such as IT devices, diagnostics, pharmaceuticals, environmental, energy industries and especially in cancer treatment[7,6].

Learning automata and grammar from strings was dealt with by researchers. D. Angulin,[3] used language learning as a central problem for computational learning theory which led to defining and studying different learning paradigms as learning with the help of oracles. Thus, the field of grammatical inference has applications in a number of research areas including machine learning, formal language theory, syntactic and structural pattern recognition, computational linguistics, computational biology and speech recognition.[2]

In this paper, we define hyperedge replacement graph grammars to generate some infinite classes of fullerene graphs and we give a learning algorithm to learn some infinite classes of fullerene graphs.

2. PRELIMINARIES

Definition 2.1:[5] Let C be an arbitrary but finite set of labels and let type: $C \rightarrow \mathbf{N}$ be a typing function. A Hypergraph H over C is a tuple (V, E, att, lab, ext) Where V is a finite set of nodes, E is a finite set of hyperedges, att : $E \rightarrow V^*$ is a mapping assigning a sequence of pair wise distinct attachment node att(e) to each $e \in E$, lab: $E \rightarrow C$ is a mapping that labels each hyperedge such that type(lab(e)) = |att(e)|, ext $\in V^*$ is a sequence of pair wise distinct external nodes. The Classes of all hypergraphs over C is denoted by H_c .

Definition 2.2:[5] A ***hyperedge replacement grammar*** is a system HRG= (N, T, P, S), Where $N \subseteq C$ is a set of non-terminals. $T \subseteq C$ with $T \cap N = \varphi$ is a set of terminals, P is a finite set of productions. A production over N is an ordered pair P = (A, R) with $A \in N$, $R \in H_c$ and type (A) = type(R). A is called the left-hand side of P and is denoted by lhs (P). R is called the right-hand side and is denoted rhs(P). $S \in N$ is a start symbol. We denote the classes of all hyperedge replacement grammars by HRG.

Definition 2.3.[5] A ***m-hypergraph*** is defined as a hypergraph with m external nodes and a handle e (single hyperedge) with attach H(e) = ext H. If labelling H(e) = A, then H is said to be handle induced by A and is denoted by A•.

Definition 2.4:[5] The ***hypergraph language*** L(HRG) generated by HRG is L_s(HRG) where for all $A \in N$. L_A(HRG) consists of all hypergraphs in H_T derivable from $A^{\bullet}$ by applying productions of P : $L_A(HRG) = \{H \in H_T / A^{\bullet} \overset{*}{\underset{P}{\rightarrow}} H\}$ We denote the classes of all hyperedge replacement grammars by HRL.

Definition 2.5:[6] A ***Fullerene graph*** is a 3-regular planar simple finite graph with pentagon or hexagon faces. In these graphs the number of pentagon faces is 12. Therefore, any fullerene graph can be characterized by number of its hexagon faces. If the number of 5-cycles (pentagons) in a given fullerene F is p and number of 6-cycles (hexagons) is h. Since each vertex lies in exactly 3 faces and each edge lies in 2 faces, then the number of vertices is

V = (5p+6h)/3, number of edges is e = (5p+6h)/2 and the number of faces is f = p + h. By the Euler's formula v − e + f = 2, one can deduce p = 12, v = 2h + 20, e = 3h + 30. For h = 0, the unique fullerene is a dodecahedron with v =20, e = 30 is given in Figure 1. There is no fullerene with h=1. The basic fullerene graphs with h=2,3,4,5,6,7 are given in figure 2.

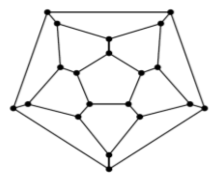


Figure 1 Fullerene h=0

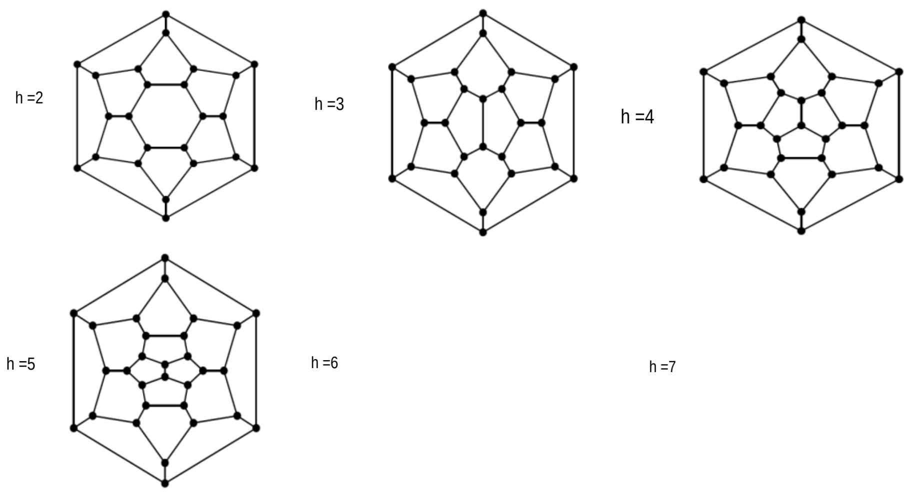


Figure 2 The Basic fullerene graphs[5]

Extending process:[6]

Let F be a fullerene with a hexagon face neighbored by pentagons only, as the above h = 2,3,4,5,7. Add a vertex to each edge of the hexagon to make all 6 neighboring pentagons, hexagon and add an edge to each new vertex, finally join the ends of new edges to make a new hexagon. With this process, we get a new fullerene with 6 more hexagons. The new fullerene has the same property and we may do the process again to get new fullerenes. We can construct fullerenes with h = i + 6k hexagons, for i = 2,3,4,5,6,7 and k = 1,2, 3... These numbers cover all-natural numbers except multiples of 6. But, the same process which is done for h = 5 to get h = 6 in Figure 2, can be done for any fullerene with h= 6k − 1 hexagons to get a fullerene with h = 6k hexagons.

3. GENERATION OF SOME INFINITE CLASS OF FULLERENE USING HYPEREDGE REPLACEMENT GRAPH GRAMMAR

The hyperedge replacement graph grammar for some infinite classes of fullerene graphs with h=i+6k, where i=2,3,4,5,6,7 is HRG= (S, D, D′, E, E′, P, S) where S is the start symbol, D, D′, E, E′ are non -terminals and P is the productions rules that are given in Figure 3.

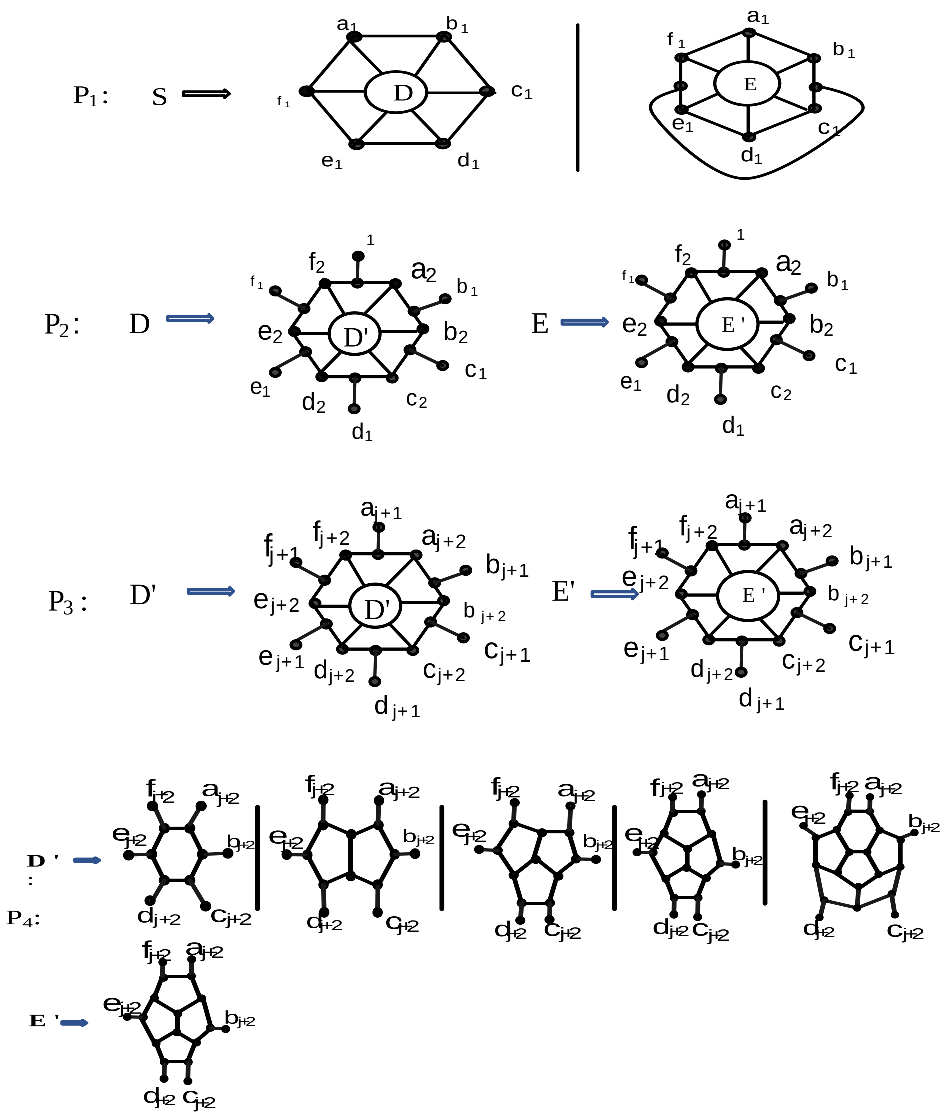


Figure 3 Production Rules that generate the infinite classes of fullerene graphs

The generation of fullerene graph with i=2 and k=1, where h=2+6.1=18 and p=12 are given in Figure 4. In this way one can generate some infinite classes of fullerene of the form h=i+6k with i=2,3,4,5,6,7 and k=1,2…

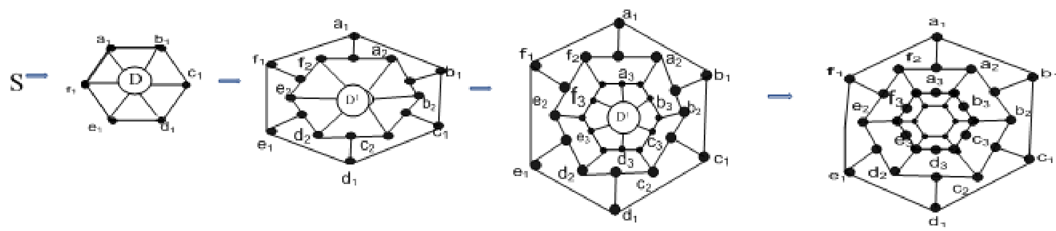


Figure 4 Generation of fullerene of the form h=i+6k, where i=2 and k=1

4. LEARNING SOME INFINITE CLASSES OF FULLERENE GRAPHS USING HYPEREDGE REPLACEMENT GRAPH GRAMMARS

Definition 4.1: ***Inner core of the Fullerene*** is obtained by removing the vertices from the outermost cycle of the basic fullerene and their corresponding adjacent vertices. There are 6 Basic Fullerene for i=2,3,4,5,6,7. They are denoted by IC(i), where i=2,3,4,5,6,7 as shown in Figure 5.

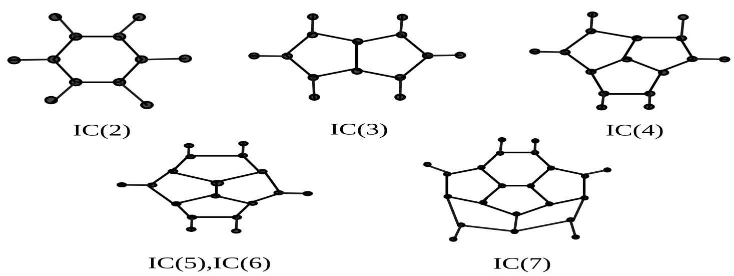


Figure 5 IC(i) where i=2,3,4,5,6,7

4.1 ALGORITHM TO LEARN SOME INFINITE CLASSES OF FULLERENE GRAPHS

This learning algorithm is the impact of Jeltsch and Kreowski work on Grammatical Inference based on Hyperedge replacement.[4].

An algorithm is given in which different classes of fullerene graphs are given as input (at least one member from each classes) along with the inner core of the fullerene. The characteristic sample is a graph with k=1 and i = 2, 3, 4, 5, 6,7. Once the Characteristic sample is taken for decomposition, the rules that generate that class is obtained.

4.1.1 ALGORITHM:

INPUT:

1. Fullerene graphs with hexagons of the form h=i+6k, where i=2,3,4,5,6,7 and k=1,2…
2. Inner core of the fullerene IC(i) i=2,3,4,5,6,7 should be initialized.

OUTPUT:

1. Production rules that generate input samples and the infinite classes of fullerene graphs of the form h=i+6k, where i=2,3,4,5,6,7 and k=1,2,3…

PROCEDURE:

Initialize Grammar Gr = {{S}, {IC(i)} (S, F_L) / L=1 to n, n ≥ 6,(S,0)$^\bullet$}
Initialize IC(i)= {IC (2), IC (3), IC (4), IC (5), IC (6), IC (7)}

Initialize Newprod=φ.

Begin

For each F_L, **do**

Introduce a new non-terminal D^j for each F_L not occurring before, j is any natural number such that **Decompose F_L** Such that

Decompose:1 Decompose(F_L) = (S, F_L^j) ∪ (D^j, F_L^{j+1}) - (S, F_L)

Newprod=Decompose (F_L)

Decompose:2, **Decompose (F_L^{j+1}),** Introduce a new non-terminal D^{j+1} not occurring before such that

Decompose (F_L^{j+1}) = (D^j, F_L^{j+2}) ∪ (D^{j+1}, F_L^{j+3})- (D^j, F_L^{j+1})

Newprod=Newprod ∪ Decompose (F_L^{j+1})

Begin

If F_L^{j+3} is equal to any of the IC(i), i=2 or 3 or 4 or 5 or6 or 7

then

Newprod =Newprod

Else

Repeat **Decompose 2:** until F_L^{j+3} is equal to any of IC(i),

End

Begin

If F_L^1 is cycle of length 6,

then

Rename NT(D^j)= {S=S,D^1=D, D^2=D^3=…=D^1}

Gr=Gr ∪ Newprod ∪ RenameNT

Else

Rename NT(D^j)= {D^1=E, D^2=D^3=…=E^1}

Gr=Gr ∪ Newprod ∪ RenameNT

End

End

Reduce is where Repeated and Redundant productions are identified and removed.
Gr = Gr-**Reduce**. The Grammar produced is given by
Gr= {S, D, D^1, E, E^1, IC(i), NewProd, S}

4.1.2 Example: Input data must contain graphs from each fullerene graph classes, For L=6, there are six input graphs given in Figure 6. The Inner Core of the basic fullerene are given as

Inputs. When DECOMPOSE 2 is applied for the first-time j = 1 and for second time j= 2 and so on.

Initialize Grammar Gr = {{S},{IC(i)}(S, F_L) / L=1 to 6 and i=2,3,4,5,6,7 (S,0)$^{\bullet}$}

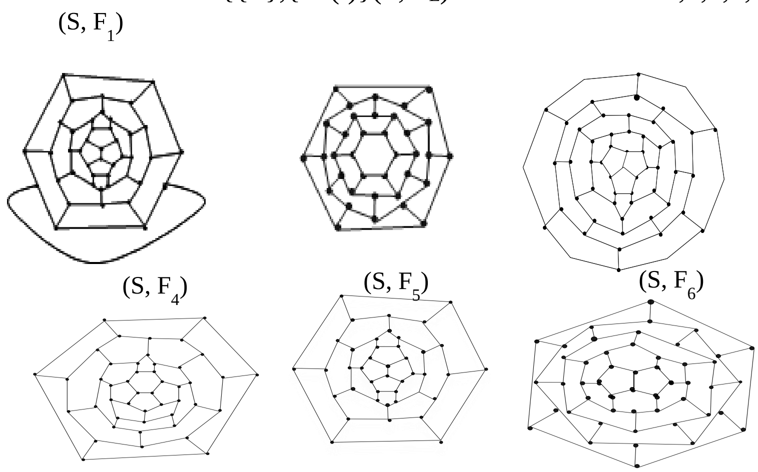
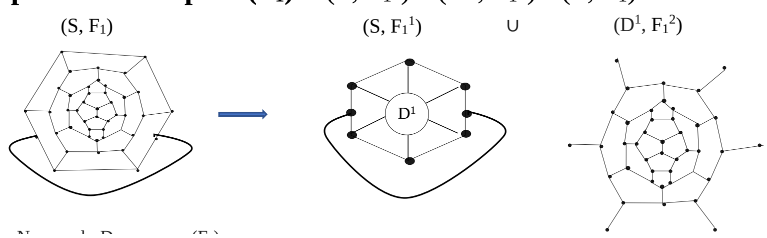


Figure 6 { (S, F_L) / L= 1 to 6 }

Initialize IC(i)= {IC (2), IC (3), IC (4), IC (5), IC (6), IC (7)}

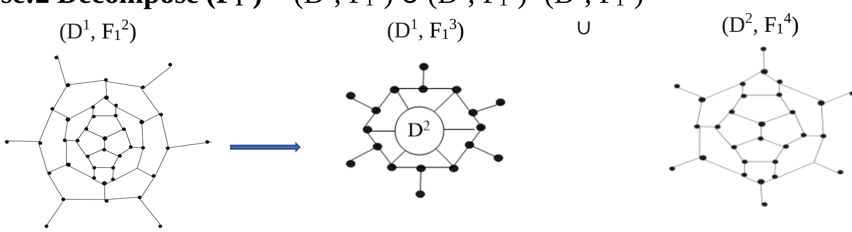
Initialize Newprod=φ.

For F_1, **Decompose:1 Decompose (F_1)** = (S, F_1^1) ∪(D^1, F_1^2) - (S, F_1)



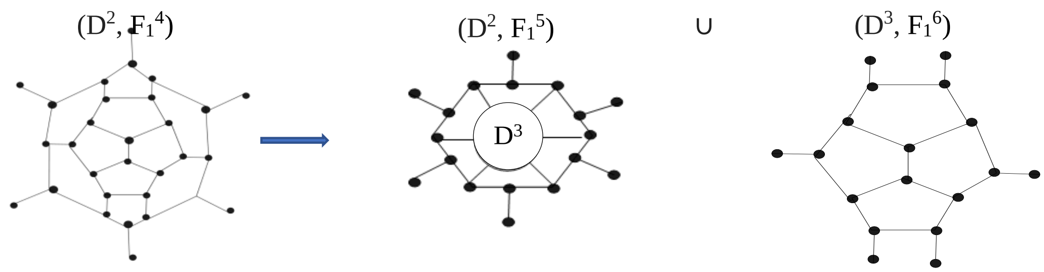
Newprod =Decompose (F_1)

Decompose:2 Decompose (F_1^2) = (D^1, F_1^3) ∪ (D^2, F_1^4)- (D^1, F_1^2)



NewProd = Decompose (F_1) ∪ Decompose (F_1^2)
F_1^4 is not equal to IC(i). Repeat, **Decompose:2**
Decompose(F_1^4) = (D^2, F_1^5) ∪ (D^3, F_1^6) - (D^2, F_1^4)

NewProd = Decompose (F_1) ∪ Decompose (F_1^2) ∪Decompose(F_1^4) and
F_1^6 is similar IC (5) =IC (6). Therefore NewProd=NewProd



F_1^1 is a not cycle of length six, **Rename NT**(D^j)= {D^1=E, D^2=D^3 =E^1}
Gr=Gr ∪ Newprod ∪ RenameNT
Similarly, For F_2, F_3, F_4, F_5, F_6 **Decompose 1, Decompose 2, Rename NT** are applied and the rules are obtained. Then **Reduce** is applied from which repeated and redundant rules are identified and removed. Gr = Gr-**Reduce**
The required grammar Gr= {S, D, D^1, E, E^1, NewProd, S} is obtained.

5. CORRECTNESS OF THE ALGORITHM[4]

1. Each sample can be derived from the axiom of an inferred grammar.
2. Each production either being an initial one or one obtained by decompositions and renaming can be used for deriving one of the samples.
3. The axiom of an inferred grammar is (S, 0)[•] or some renaming of it because the axiom has this form initially and the RENAME operation is the only one affecting the axiom. A grammar with the above properties is called samples composing. Hence our grammar is samples composing as it satisfies the above conditions.

Since the HRG of infinite classes of fullerene graphs is a subclass of the classes of hyperedge replacement grammars in[4], it is decidable, that some infinite classes of fullerene graphs can be inferred.

6. CONCLUSION

We have generated some infinite classes of fullerene graphs using hyperedge replacement graph grammars and a learning algorithm is discussed. It can be extended to other allotropes of carbon, which has wide applications in medical field.

REFERENCES

1. A.Habel, H.-J. Kreowski: May We introduce To you:Hyperedge Replacement, *Lecture Notes in Computer Science*, 291,15-26(1987).

2. Colin de la Higuera, Current Trends in Grammatical Inference, *Lecture Notes in Computer Science*, 1876,28-31 (2000).
3. D. Angulin, On the Complexity of Minimum Inference of Regular Sets. *Information and Control*, 39,337-350 (1978).
4. E.Jeltsch, H.Kreowski, Grammatical Inference based on Hyperedge Replacement, *Lecture Notes in Computer Science*, 532, 461-474(1990).
5. G.Rozenberg, Handbook of Graph Grammars and Computing by Graph Transformation, *World Scientific* (1997).
6. Rashid Zaare-Nahandi, Some Infinite Classes of Fullerene Graphs International Mathematical Forum, 1, 40 (2006).
7. Vesna Andova, Frantiˇsek Kardoˇs, Riste Skrekovski,Fullerene graphs and Some Relevant Graph InvariantsTopics in Chemical Graph Theory,39-54 (2014).