Integrated Intelligent Research (IIR)

International Journal of Computing Algorithm
Volume: 03, Issue: 01 June 2014, Pages: 58-60
ISSN: 2278-2397

DNA Algorithm for Pancyclicityand Vertex Connectivity of Graph

Antony Xavier, Andrew Arokiaraj
Department of Mathematics, Loyola College, Chennai , India
Email: andrewarokiaraj@gmail.com

Abstract- Finding the Pancyclicity and Vertex connectivity for a general graph is the problem in NP complete class. In this paper, a bio-computation way of solving these two problems has been proposed. DNA computation is more powerful because of its massive parallelism and high density storage capacity. Thereby, producing the VERTEXCONNECTIVITY, PANCYCLIC and GIRTH algorithm solvable in polynomial time. The method in tracing the k-cycle and k-cut vertex was also used for the computation.

Keywords: DNA algorithm, pancyclic, Vertex connectivity, Girth.

I. INTRODUCTION

DNA computation is an emerging Bio-computing model in solving most of the NP complete problems, and making them solvable in polynomial time. The seminal work was done by Adleman in solving Hamiltonian path problem and Lipton solving the satisfiability problem. This new computational method has more parallelism and high storage capacity, which brings enormous break-through in theory of computation, graph theory, cryptography. Further both these computing ideas were extended in solving graph theoretical problems like TSP, coloring, maximum clique, isomorphism, vertex cover, etc.In this paper we use DNA to compute the pancyclicity, garth, vertex connectivity of the graph. Fo this we imbibe the style which Adleman used in solving computational problems with DNA. The various DNA operators used in the algorithm were also explained.

II. GRAPH PROBLEMS

The following the graph theoretical problems, for which the DNA algorithm in polynomial time have been suggested.
Vertex Connectivity:
The minimum number of nodes whose deletion froma graph G disconnectsit. Vertex connectivity is sometimes called "point connectivity" or simply "connectivity."
Pancyclic:
An n-vertex graph G is pancyclic if, for every k in the range $3 \leq k \leq n$, G contains a cycle of length k.
Girth:
The girth of a graph is the length of a shortest cycle contained in the graph

III. MOLECULAR BIOLOGY

DNA (Deoxyribonuclic acid) is double stranded helical in structure. Each strand is a chain of nucleotide. The four nucleotides are adenine (A), guanine (G), Thymine (T), cytosine (C). DNA obeys the following complementality: A always pairs with T and G always pairs with C know as Watson-Crick complementarities.

DNA IN CODING OF A PATH
The idea of encoding a graph path by DNA was introduced by Adleman. Consider a graph in fig 1.

Each of its vertex is represented by a string of length ‘n’ using combination of A,G,T,C with m=2,let v1=AA, v2=AC,v3=TT,v4=CC,v5=AT. A path is given by double stranded helix

IV. STICKER BASED DNA COMPUTATION

The sticker model was introduced by S. Roweis et al. In this model, there is a memory strand with N bases in length subdivided into K non-overlapping regions each M bases long (N > MK). M can be for example 20. The substrands (bit regions) are significantly different from each other. One sticker is designed for each subregion; each sticker has M bases long and is complementary to one and only one of the K memory regions. If a sticker is annealed to its corresponding region on memory strand, then the particular region is said to be on. If no sticker is annealed to a region then the corresponding bit is off. Each memory strand along with its annealed stickers is called memory complex. In sticker model, a tube is a collection of memory complexes, composed of large number of identical memory strands each of which has stickers annealed only at the required bit positions. This method of representation of information differs from other methods in which the presence or absence of a particular subsequence in a strand corresponded to a particular bit being on or off. In sticker model, each possible bit string is represented by a unique association of memory strands and stickers. This model has a random access memory that requires no strand extension and uses no enzymes.

STICKER OPERATIONS:

- separate (T, T+, T-,i): Consider a test tube T and an integer i and create two new test tubes T+ and T-, where T+ consists of all memory complexes in T in which ithsubstrand is on, while T- is comprised of all memory complexes in T in which the ithsubstrand if off.
- incident(i,j): Checks the existence for ithvertx and jth vertex to be adjacent to each other.
- merge(T1,T2,…,Tn): Take tube T1,…,Tn, produce their union T1 U T2 U … U Tn, put the result into the (possibly empty) tube Tn.

International Journal of Computing Algorithm
Volume: 03, Issue: 01 June 2014, Pages: 58-60
ISSN: 2278-2397

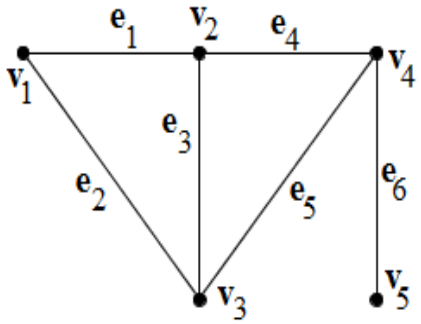
- discard (T_0): Take a test tube T and empty its contents.
- set (T-, n+m+i_j): Start with a test tube T and an integer i and generate a test tube in which the ithsubstrand of each memory complex is turned on.
- clear (T++, n+m+i_j): Start with a test tube T and an integer i and generate a test tube in which the ithsubstrand of each memory complex is turned on.

V. SUGGESTED ALGORITHM

The initial test tube T0 contains generated random paths of Graph G. This is achieved by annealing and ligation. The time complexity for this generation is taken as 1 unit because of immense parallelism.

The algorithm INCIDENTRELATION provides the incidence relation between vertices and edges of a graph. The input of the algorithm is an $[m+n\ ;\binom{n}{k}]$ library T, providing encoded DNA of all *k*-subsets of vertices, where $1<k<n$. Two parameters lower (*l*) and upper (*u*) bound on the set of strands of size *n*, where $1<l<u<n$. For those memory complexes whose i^{th}substrands is turned on, $1<i<u$, the algorithm verifies in parallel if the vertex-edge pair (v_i, e_j) is incident and if so, turns on the $u+j^{th}$substrand corresponding to the incident edge (statements 4-5). At the end of the loop, the strands composed of the last *m* substrands provide the incidence pairs (v_i, e_j) between vertices and edges. The algorithm requires $n(m+2)$ steps.

INCIDENTRELATION(T,*l*,*u*,*m*,*n*)
Input: $[m+n\ ;\binom{n}{k}]$ library T, $1<l<u<n$
01. for*i*←l to *u*do
02. separate (T, T+, T-,*i*)
03. for*j*←1 to *m*do
04. if incident(*i*,*j*) then
05. set (T+, *u*+*j*)
06. end if
07. end for
08. merge(T+,T-,T)
09. end for
10. return T
l=1 and u=5



v_1	v_2	v_3	v_4	v_5	e_1	e_2	e_3	e_4	e_5	e_6
1	0	0	0	0	1	1	0	0	0	0
0	1	0	0	0	1	0	1	1	0	0
0	0	1	0	0	0	1	1	0	1	0
0	0	0	1	0	0	0	0	1	1	1
0	0	0	0	1	0	0	0	0	0	1

The algorithm WEIGHTENING extracts from an input test tube T_0 those memory complexes in which exactly k of the sunstrands m+1,…,m+n are turned on, where $0<k<n$. At the end of the loop (1-7), the test tube T_i, where $0<i<n$, contains all memory complexes in which exactly i of the substrands m+1,…,m+n are turned on. Thus the test tube T_k provides the output of the algorithm. The sticker algorithm requires $2n[(n+1)/2] = n^2+n$ steps.

WEIGHTENING (T_0, *m*, *n*, *k*)
Input: input test tube T_0
01. for*i*←0 to *n*-1 do
02. for*j*←*i* down to 0 do
03. separate (T_j, T+, T-, *m*+*i*+1)
04. merge (T+, T_{j+1})
05. merge (T-, T_j)
06. end for
07. end for
08. returnT_k

Consider an input tube T_0 providing encoded DNA of the memory complexes 0001, 0110, 1011. The computation of weightening for m=0 and n=4 is as follows.

	T_0	T_1	T_2	T_3	T_4
Initial	0001 0110 1011				
i=1	0001 0110	1011			
i=2	0001	0110 1011			
i=3	0001		0110 1011		
i=4		0001	0110	1011	

K-CYCLE (T,*m*,*n*,*k*)
Input: $[m+n\ ;\binom{m}{k}]$ library T
01. T ←EDGEINDUCEDGRAPHS(T,*m*,*n*,*k*)
02. T ← WEIGHTENING (T,*m*,*n*,*k*)
03. for*i*← 1 to *m*do
04. separate (T, T+, T-, *i*)
05. for*j*← 1 to 2 do
06. separate (T+, T++, T+-, $n+m+i_j$)
07. set (T+-, $n+m+i_j$)
08. clear (T++, $n+m+i_j$)
09. end for
10. end for
11. T ← WEIGHTENING (T,*m*+*n*,*n*,*0*)
12. if ┐empty (T) then
13. return (T)
14. else
15. report “No k-cycle”
16. end if

GIRTH (T,*m*,*n*)
Input: $[n+(m+n)\ ;\ \binom{n}{k}]$ library T
01. for*k*← 3 to *n*do
02. T_0← K-CYCLE (T,*m*,*n*,*k*)
03. if ┐empty (T_0) then
04. return (T_0)

International Journal of Computing Algorithm
Volume: 03, Issue: 01 June 2014, Pages: 58-60
ISSN: 2278-2397

```
05.     end if
06. end for

PANCYCLIC (T,m,n)
Input: [n+(m+n) ; (n k)] library T
01. fork← 3 to ndo
02.     T0← K-CYCLE (T,m,n,k)
03.     if empty (T0) then
04.             report "Not Pancyclic"
05.     end if
06.     discard (T1)
07. end for
08. report "Graph is Pancyclic"

K-CUTVERTEX (T,m,n,k)
Input: [2n ;(n k)] library T
01. fori← 1 to n-1do
02.     separate (T, T+, T-, i)
03.     set (T-, n+i)
04.     forp←n+ i to 2n-1do
05.             separate (T-, T-+, T--, p)
06.             forq←2n down to n+i+1do
07.                     if adjacency (p,q) then
08.                             set (T-+, q)
09.                     end if
10.             end for
11.             forj←1 to ndo
12.             separate (T-+, T-++, T-+-, j)
13.                     clear (T-++, n+j)
14.                     merge (T-++, T-+-, T-+)
15.             end for
16.             merge (T-+,T--,T-)
17.     end for
18.     merge (T-, T0)
19.     merge (T+, T)
20. end for
21. T ← WEIGHTENING (T, n+1, n, k+1)
22. if ┐empty (T) then
23.     report "k-cut vertex"
24.     return (T)
25. else
26.     report "No k-cut vertex"
27. end if

VERTEXCONNECTIVITY (T,m,n)
Input: [2n ;(n k)] library T
01. fork← 2 to n-2do
02.     T ← K-CUTVERTEX (T,m,n,k)
03.     if ┐empty (T1) then
04.             report "k-vertex connectivity"
05.             return (T1)
06.     end if
07. end for
```

VI. CONCLUSION

This paper demonstrates that Wiener Index can be solved using DNA computation. This method suggested can be applied for general graph in finding Girth, Pancyclic and Vertex Connectivity, which are NP-complete problems. The complexity of these algorithm in worse case yields $O(n^2)$. Also, the method in tracing the shortest path was also discussed within the algorithm.

REFERENCES

[1] Ahmed Al-Kandari, Paul Manuel, IndraRajasingh. Wiener Index of Certain Interconnection Networks.International Conference of Mathematical and Computer Science ICMCS 2011.
[2] BojanMohar and Tom Pisanski.How to Compute the Wiener Index of a Graph. Journal of Mathematical Chemistry 2(1988)267-277.
[3] J in Xu, Xiao Ii Qiang, Kai Zhang, Cheng Zhang, J ing Yang, Rongkui Zhang. A parallel type of DNA computing model for graph vertex coloring problem.IEEE 2010.
[4] Leonard Adleman. Computing with DNA. Scientific American, 279(2):54– 61, August 1998
[5] Martyn Amos. Theoretical and Experimental DNA Computation.Springer. 2005.
[6] Mehdizadeh, Nekoui, Sabahil, Akbarimajd. A Modified DNA-Computing Algorithm To Solve TSP. IEEE 2006.
[7] Mehdi Eatemadi, Ali Etemadi, & Mohammad-Mehdi Ebadzadeh. Finding the isomorphic graph with the use of algorithm based on DNA. International Journal of Advanced Computer Science, Vol. 1, No. 3, Pp. 106-109, Sep. 2011.
[8] Richard J. Lipton. DNA Solution of Hard Computational Problems. Science, New Series, Vol. 268. : 542-545, April 1995
[9] Yang Jing, Zhang Cheng, Xu Jin, Liu XiangRong&QiangXiaoLi. A novel computing model of the maximum clique problem based on circular DNA. Science China Press and Springer-Verlag Berlin Heidelberg 2010
[10] Xuncai Zhang, Ying Niu, Fei Li, ZuoxinGan. Solving minimum vertex cover problems with microfluidic DNA computer.IEEE 2012.
[11] ZoyaIgnatova, Israel Martinez, and Karl-Heinz Zimmermann.