Egyptian Informatics Journal 24 (2023) 173–190

Contents lists available at ScienceDirect

Egyptian Informatics Journal

journal homepage: www.sciencedirect.com

Intelligent deep analysis of DNA sequences based on FFGM to enhancement the performance and reduce the computation

Zena A. Kadhuim [a], Samaher Al-Janabi [b],*

[a] Department of Software/College of Information Technology, University of Babylon, Babylon, Iraq
[b] Department of Computer Science, Faculty of Science for Women (SCIW), University of Babylon, Babylon, Iraq

ARTICLE INFO

Article history:
Received 6 May 2022
Revised 21 February 2023
Accepted 24 February 2023
Available online 7 March 2023

Keywords:
Intelligent deep analysis
DNA sequence
RNA sequence
Graph Mining Techniques
FFGM

ABSTRACT

In an attempt to improve the analysis DNA sequence, a new intelligent deep analysis algorithm called reduce frequency bast on fast frequency graph mining (RF-FFGM) is established; This algorithm at the beginning converts the DNA sequence into RNA sequences after that split these sequence into multi subsequence through determined specific equation for start and end point of each sequence. After that each subsequence represent as subgraph after label to the bonds between each pair of components related to RNA (i.e., A, G, U, C) these bounds include 16 labels used as Knowledge Constructions (KC)) for this work. After that apply the steps of FFGM that select after deep analysis to graph mining techniques ***(GSpan, FFSM, Hybrid-Tree-Miner, Approximate Frequent Sub-graph, CloGraMi and FFSM)*** **this analysis focus on determined *(the main programming steps, main parameters, advantages, disadvantages) for each algorithm.*** We discovery FFGM finds the frequent in a short time, because it building matrix code for connection edge and transforming matrices into incidence matrix, also; we found FFGM can get all the edges that have the highest contact with the other edges, so from the second stage therefore it avoids us from going through a sequential path to find duplicate edges. RF-FFGM appears as a pragmatic algorithm, it proves their robust to work with DNA sequence to reduce the computation and time.

1. Introduction

Deoxyribonucleic acid (DNA), is a good potential for understand the human genome of society. The major benefit from studying DNA is to detect and analysis the genome structure of living organisms. DNA represented by four basic elements called base or nucleate, these elements are Adenine (A), Thymine (T), Guanine (G) and Cytosine (C) [22]. DNA can be analysis using many software and tools available called Basic Local Alignment Search, Tools (BLAST). it was used to scan sequences, from more than 260,000 species, including more than 190 billion nucleotides [19,25].

Data is important for knowing knowledge, it can be defined as an information that have been converted to a suitable form for processing, in other word data is a collection of objects and their attribute related to it or object with collection of features or object with a specific single feature that recognize it [24]. It comes in several forms and get by observation, search, or recording. While the data science is integration among three domain (statistical or mathematical, computer science and knowledge) domain [7]. Small, normal, and big data are three type of data science; small data organized in uniform structure such as list or a table, and the size of data no greater than 30 samples, and not submitted to the normal distribution and cannot be, used to make any decisions; normal data, on the other hand, is structured data that is submitted to, the normal distribution and can be used to make various decisions, such as (clustering, classification, prediction and optimization, etc.). Finally, large data, which can be structured, semi structured, or unstructured, has a size range of 1 TB to 1 ZB. It can extract important information or pattern it using a mix ML and Cloud computing [20]. Different type data science have been introduced with their used in different field and advantage of it. Most programmers used normal data because of simplicity in gen-

Abbreviations: A, Adenine; AP-FSM, Approximate frequent sub-graph; C, Cytosine; CloGraMi, Closed Frequent Sub-graph Mining; Close, Close frequent edge; DNA, Deoxyribonucleic acid; M(S), Minimum supported threshold; MF(S), Maximum frequent edge; mRNA, Messenger Ribonucleic acid; FFSM, Frequent, subgraph mining; GSpan, Graph based substructure pattern mining; G, Guanine; RF-FFGM, Remove duplication of frequency based on fast frequency graph mining; T, Thymine; U, Uracil.

* Corresponding author.

E-mail address: samaher@itnet.uobabylon.edu.iq (S. Al-Janabi).

https://doi.org/10.1016/j.eij.2023.02.004
1110-8665/© 2023 THE AUTHORS. Published by Elsevier BV on behalf of Faculty of Computers and Artificial Intelligence, Cairo University.

eration model and move away from using cloud computing. The term of Finding, gathering, cleaning, evaluating, and modelling data in order obtain relevant information and insights and interpret the derived information, for data-driven decision-making is refer to data analysis [11,13].

It is hard to understanding the structure of DNA, includes protein structure or any chemical structure. DNA represented in form of graph as a large collection of nodes and edge. There is a lot of search algorithm in graph have been used recently [22], graph mining has become an active, and important in data mining, The most basic frequent patterns that may be detected in a collection of graphs are known as frequent substructures. They can be used to characterize graph sets, distinguish between distinct groups of graphs, categorize and cluster graphs, generate graph indices, and make similarity searches in graph databases easier. Several graph mining approaches have recently been created and applied to the detection of intriguing patterns in a variety of applications [2].

The reminder of this paper organized as followed: section two the related work that people research in the same field, section three represents the important of graph mining and algorithm related to it. Section four represents the methodology of our paper; section five shows the implementation of our work while section six present the result of our method finally section seven is conclusion and future work.

2. Related work

Mining of frequency sub graph is one of the most important concerns that directly effect on computation of all sequence later As a result, in this section of the paper, we will attempt to review previous research in the same field as our topic and compare these works in terms of five essential characteristics, namely the database/ dataset employed. The methods for evaluating the findings, as well as the benefits and drawbacks of each method, are discussed, as well as evaluation.

Lopamudra et al.[16] built different ML model in order to predict Protein -Protein Interaction between human protein and viruses to discover anti-COVID drug, first must prepare model to accept various length of Protein human sequences. For preprocessing data they used LVQ for feature subset selection and then used different supervised learning algorithm (SVM), (NB), (RF), (KNN) with multilayer perceptron used for prediction and classification. For performance measurers they used confusion matrix for evaluating the performance of prediction. Our method similar with this method for preparing sequence of structure but different in technology used based on Frequency sub-graph mining algorithm.

Xiujuan et al. [27] design a new method called Deep learning based protein-binding site prediction with feature based non-redundancy from level of RNA (DFpin) for predicting protein interaction related to different type of human Disease, they used sliding window, and for each window apply preprocessing technique, they Remove feature redundancy based on the RNA mono-nucleotide composition to maintain the diversity, of RNA samples and avoid the residue of redundant data then they used SVM to identify PIN of protein interaction related to varies type of protein, for dataset used they used different length of RNA dataset: RNA587, RNA208 for predict protein interaction. Our method similar with this method for preparing sequence of structure but different in technology used based on Frequency sub-graph mining algorithm.

Chang [4] built deep new learning mechanism named (F UTUSA) to predict protein function based on information of sequence of RNA only, First sequence is segmented in order to improve the prediction of protein function by a CNN, for train the regional sequence patterns and their relationship. FUTUSA also performed admirably in terms of predicting acetyl transferase and demethylase activity. The next step was to see if FUTUSA could anticipate the functional consequences of a point mutation. FUTUSA was successful after being trained for monooxygenase activity, predicted the effect of point mutations on the enzyme phenylalanine hydroxylase, which causes an inherited condition. KU is a metabolic disorder. This deep-learning algorithm can be used to characterize newly discovered objects as a first step. Proteins that haven't been identified or that haven't been well examined. For performance measurers they used accuracy for evaluating the performance of prediction. Our work similarity with this work in the idea of predicting protein for identify which family cased disease and which is stop it, and in evaluation measurement but we use confusion matrix include not only accuracy but differ in method used to discover protein based on intelligent data analysis technique. Our method similar with this method for preparing sequence of structure based segmentations of mRNA sequence but different in technology used based on Frequency sub-graph mining algorithm.

Minghui Wang et al. [17] design a new model called Malsite-Deep for prediction malonylation sites that protein occurs in response to lipopolysaccharide (LPD). At the first stage apply seven feature extraction methods to protein to extract feature information from it. Second stage handle imbalance data by applying the under-sampling NearMiss-2 and sellect best feature subset. Last stage these data go to Deep Neural Networks predict sites of malonylation. Evaluation of model by 10-fold cross model performance is evaluated by 10-fold validation and independent test sets shows that the AUC value on the training dataset reaches 0.99. The AUC values on the four independent test datasets all reach above 0.95. Our method similar with this method for preparing sequence of structure but different in technology used based on Frequency sub-graph mining algorithm.

Zicheng et al. [28] design method to identify immune-related Retinol-binding proteins (RBP) are a family of proteins with diverse functions to predict prognosis and therapy response in prostate cancer. Before work must segment the mRNA sequence to improve work. They used mRNA dataset and used Pearson Correlation analysis for select Immune-related RBP and they found this method able to distinguish among the high-risk and low-risk groups of prostate cancer. high-risk suffer from higher rates of genomic alterations, and were more sensitive to targeted and immunotherapy, than the low-risk group by using sensitivity measurement. Our method similar with this method for preparing sequence of structure based segmentations of mRNA sequence but different in technology used based on Frequency sub-graph mining algorithm.

3. Graph mining

A well-establish research field, in computer science and many other field of science is a Data mining. It can be found in many different area such as text [23], image mining [9], sound mining [3], video mining [26] and graph mining [5].

Recently, graph mining used in many area such as Chemical, Compound, Structure of Protein, Social Network and Flow of Program [8]. It allows to analyze, process, discover meaningful knowledge from graph data. In general, graph mining is the process of extracting non-trivial graph structures from a single graph or a set of graphs. The following is a definition of frequent sub-graph mining, which is a common example of graph mining difficulties. Find all sub-graphs whose support is no less than a user-specified minimum supported threshold, given a graph data set D = G1, G2,. . ., Gn, where Gi = (Vi, Ei) (1 I n) is a graph with a vertex set Vi and an edge set Ei, and the support of a sub-graph g is the number of graphs in D that g is sub-graph isomorphic to [5]. Frequent sub-graph mining In graph mining is one of the most challenging tasks of graph mining. Frequent sub-graph mining divided into two important strategies first, FSM based a priori approaches that used for transactional data and small graph while second, FSM based pattern growth approaches for largest graph or multiple graph[21].

The first algorithm is the basic one algorithm used for extracting all frequent subsequence / sub-graph from data and then accepting the most frequent subsequence according to some minimum support. Apriori algorithm generate sub-graph of, size-(k + 1) of k sub-graph and uses breadth first,search strategy for visiting nodes. FSG and AGM is the most used Apriori algorithm it used edge based method for growing the size of sub-graph by adding one edge for each iteration [2]. The second algorithm used for extracting all frequent sub-graph from data and then accepting the most frequent subsequence according to some minimum support. Pattern growth algorithm uses depth first,search strategy for visiting nodes. gSpan, FFSM, AP- FSM, Hybrid-Tree-Miner and CloGraMi is the most used pattern growth algorithm and used edge based method for growing the size of sub-graph by adding one edge for each iteration [11,13].

3.1. Graph based substructure pattern mining (GSpan)

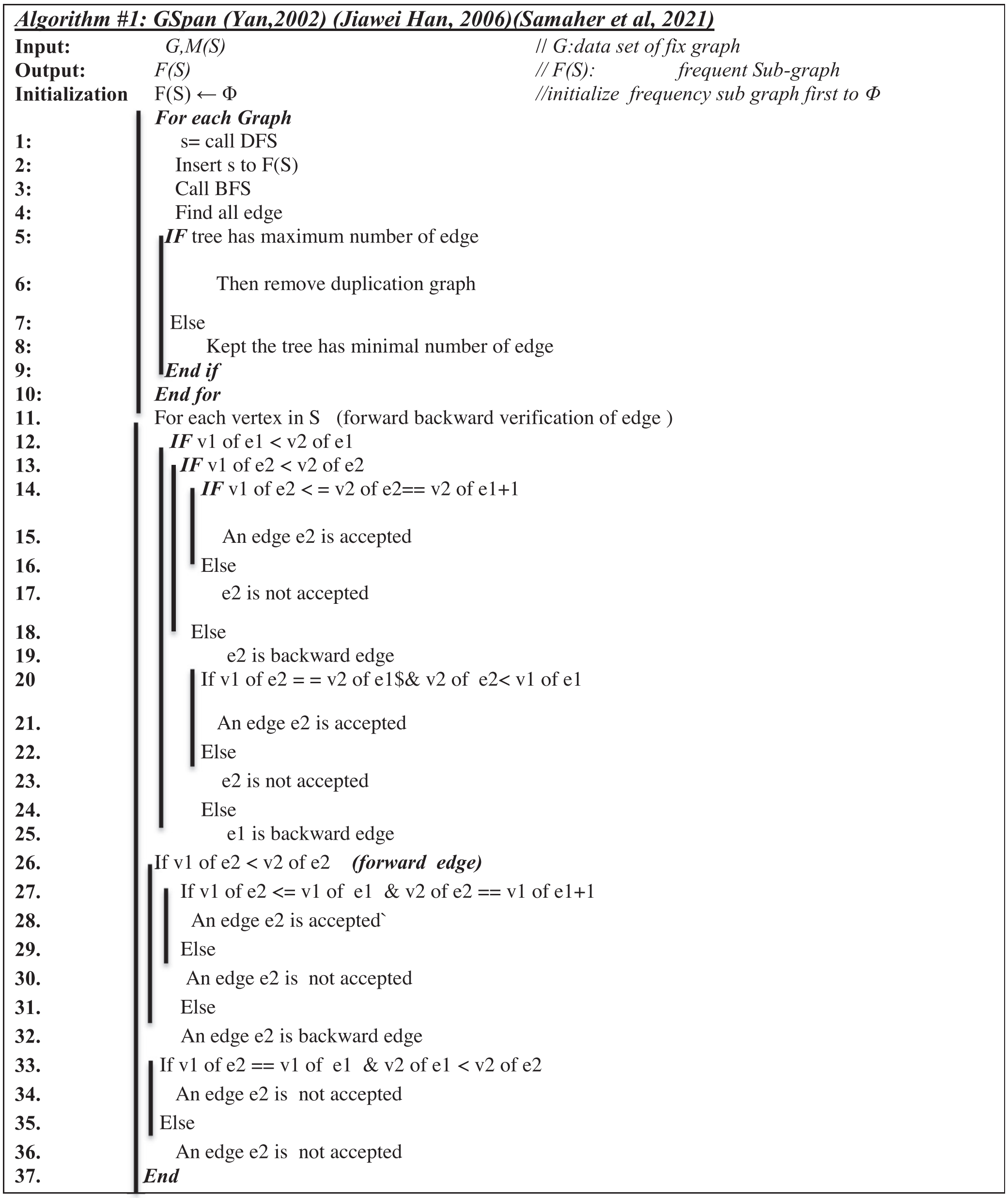
Graph based substructure pattern mining, (GSpan) is a complet frequent sub-graph mining, work on large fix graph and also labeled graph, GSpan uses pattern growth, strategy for extension the graph, by adding new edge to generate set of candidate, sub-graph and using DFS for building frequent sub-graph bottom up, then remove redundancy by minimal DFS cods according to lexical ordering. The performance of this, algorithm is improved over Apriori extensions to graphs through DFS Code representation and candidate pruning [20]. GSpan Discovered by Yan, X., & Han, J. in 2002, And publish this algorithm in Journal of Chemical, Information and Modeling (Yan, 2002). main goal of GSpan is to finding all frequent sub-graphs without candidate generation frequent sub-graph using DFS and false positives pruning, and also reduces duplicate, graph generation. The main parameters of GSpan algorithm is G: represent a large fix graph contain set of edge e and nodes n, M(S): that represent minimum supported threshold of selected sub-graph desire. Algorithm bellow discusses how GSpan based Frequent Sub-graph mining work. Algorithm 1 show GSpan.

3.2. Frequent, sub-graph mining (FFSM)

To reduce computation of counting frequent sub-graph, another algorithm based graph is represented Frequent, Sub-graph Mining (FSM). In this algorithm (FSM) represent graph in canonical adjacency matrix (CAM), for each edge and vertices of the graph and uses a vertical search technique within an algebraic graph structure. FSM outperforms the current state-of-the-art sub-graph mining technique GSpan by a significant margin. The FSM discovered by Huan[10] and published in UNC computer science technique. And later modified and publish in IEEE. FSM outperforms GSpan and Apriori techniques by efficiently tackling the time-consuming underlying sub-graph isomorphism problem and proposing two efficient sub-graph enumeration operations, as well as an algebraic graph framework built to limit the number of duplicate candidates presented [10]. The main parameters of FSM algorithm is G: represent a large fix graph and M(S): that represent minimum supported threshold of selected sub-graph desire and also Sa the canonical adjacency matrix of the frequent node, Pa canonical adjacency matrix of the frequent edge. The algorithm bellow discuss how FSM based Frequent Sub-graph mining work. Algorithm 2 shown Frequent, Sub-graph Mining.

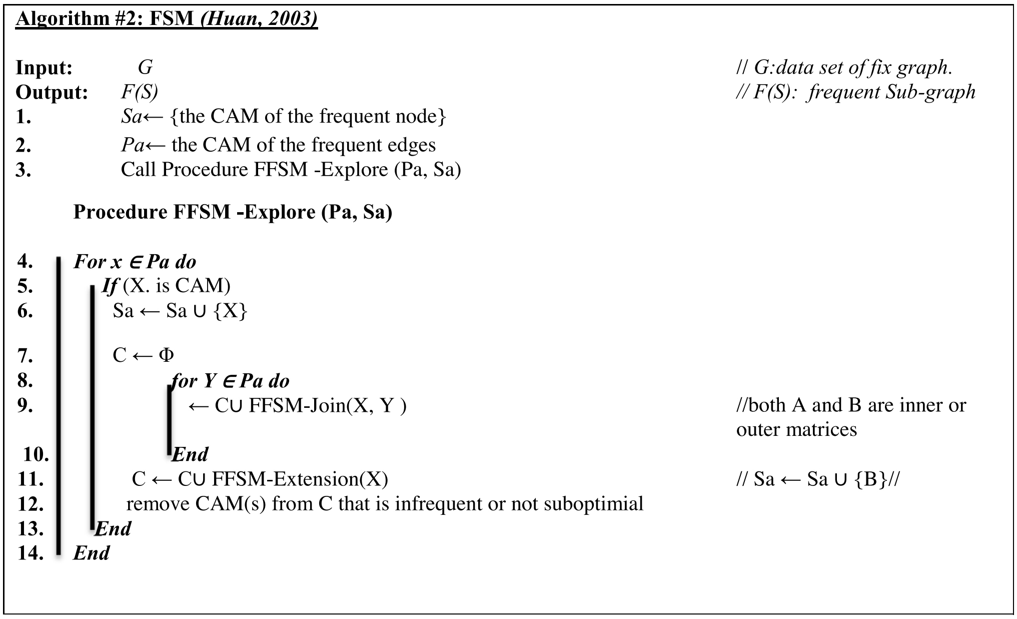
3.3. Hybrid-Tree-Miner

Hybrid-Tree-Miner is a computationally efficient program that finds all frequently occurring patterns in a tree. Sub-trees in a rooted unordered tree database by traversing an enumeration, the method finds frequent sub-trees. A tree that counts all sub-trees in a systematic manner The enumeration tree is based on a new canonical model. The breadth-first canonical form for rooted unordered trees. This approach can efficiently handle databases of free trees by extending the definitions of our canonical form and enumeration tree to free trees. Extensive tests on this algorithm based on both synthetic data and datasets but used in static graph from real-world examples. Algorithm discovered by [6]. As a result, this algorithm of finding frequent subtree mining approach works on both rooted and unrooted trees. Rooted unordered trees and free trees have similar canonical forms, enumeration trees, operations on the enumeration trees, and frequent subtree mining strategies in our construction [6].



Algorithm #1: GSpan (Yan,2002) (Jiawei Han, 2006)(Samaher et al, 2021)

```
Input:           G,M(S)          // G:data set of fix graph
Output:          F(S)            // F(S):        frequent Sub-graph
Initialization   F(S) ← Φ        //initialize  frequency sub graph first to Φ
For each Graph
    s= call DFS
    Insert s to F(S)
    Call BFS
    Find all edge
    IF tree has maximum number of edge
        Then remove duplication graph
    Else
        Kept the tree has minimal number of edge
    End if
End for
For each vertex in S   (forward backward verification of edge )
    IF v1 of e1 < v2 of e1
        IF v1 of e2 < v2 of e2
            IF v1 of e2 < = v2 of e2== v2 of e1+1
                An edge e2 is accepted
            Else
                e2 is not accepted
        Else
            e2 is backward edge
            If v1 of e2 = = v2 of e1$& v2 of  e2< v1 of e1
                An edge e2 is accepted
            Else
                e2 is not accepted
        Else
            e1 is backward edge
    If v1 of e2 < v2 of e2    (forward  edge)
        If v1 of e2 <= v1 of  e1  & v2 of e2 == v1 of e1+1
            An edge e2 is accepted`
        Else
            An edge e2 is  not accepted
        Else
        An edge e2 is backward edge
    If v1 of e2 == v1 of  e1  & v2 of e1 < v2 of e2
        An edge e2 is  not accepted
    Else
        An edge e2 is  not accepted
End
```



Algorithm #2: FSM *(Huan, 2003)*

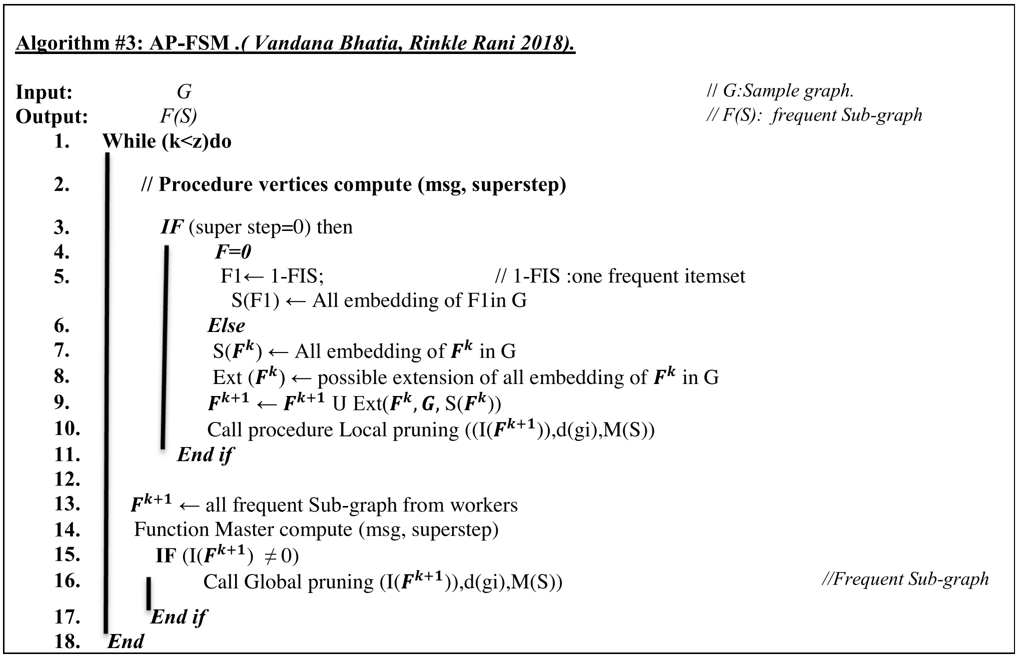
```
Input:    G                                                    // G:data set of fix graph.
Output:   F(S)                                                 // F(S):  frequent Sub-graph
          Sa← {the CAM of the frequent node}
          Pa← the CAM of the frequent edges
          Call Procedure FFSM -Explore (Pa, Sa)

     Procedure FFSM -Explore (Pa, Sa)

     For x ∈ Pa do
         If (X. is CAM)
           Sa ← Sa ∪ {X}

           C ← Φ
                 for Y ∈ Pa do
                   ← C∪ FFSM-Join(X, Y )                       //both A and B are inner or
                                                               outer matrices
                 End
             C ← C∪ FFSM-Extension(X)                          // Sa ← Sa ∪ {B}//
             remove CAM(s) from C that is infrequent or not suboptimial
         End
     End
```

3.4. Approximate frequent sub-graph AP-FSM

The majority of available frequent sub-graph mining techniques are centralized algorithms that can't efficiently manage a single huge graph and have substantial communication costs. However Pregl, a distributed graph environment, was used to implement Ap-FSM. Ap-operation FSM's is divided into two parts. There are three stages. The first step selects a representative graph from the original graph while maintaining the original graph's integrity. The qualities of the original graph Sub-graph extension is efficiently performed in the second step. In Phase 3, a novel two-step optimization for performing sub-graph trimming is introduced. Graph data of this size is difficult to analyze. Vandana Bhatia and Rinkle Rani [2] and published at Expert Systems with Applications in Elsevier journal. Approximate sub-graph mining can be used to reduce the computing cost of sub-graph mining by capturing similar structural sub-graphs. As far as accurate sub-graph mining is concerned. Will be advantageous from the standpoint of an expert and intelligent systems, as discovered patterns can be used for knowledge discovery and decision making. Vandana Bhatia, Rinkle Rani [2]. The main parameters of AP-FSM algorithm is G: represent a sample graph and M(S): that represent minimum supported threshold of selected sub-graph desire and also Sa the adjacency matrix of the frequent node, Pa adjacency matrix of the frequent edge. Algorithm 3 shown AP-FSM. Algorithm 3 shown Ap-FSM while 3.A shown execution and 3.B AND 3.C shown the local and global of that algorithm.

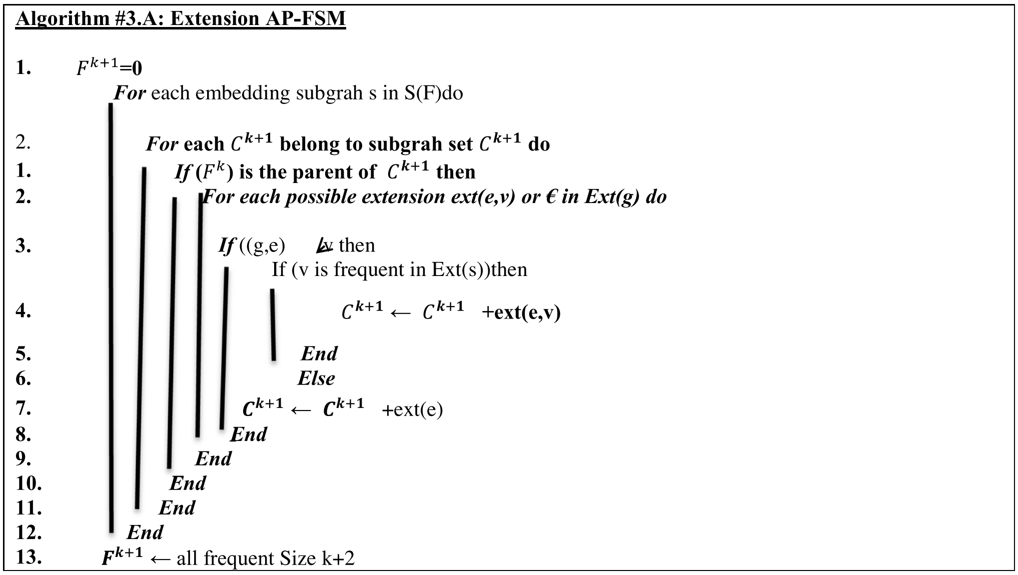
3.5. CloGraMi

CloGraMi (Closed Frequent Sub-graph Mining) is a GraMi-based approach for locating all closed frequent sub-graphs in a big graph. The first is a new level order traversal approach for quickly determining closed sub-graphs in the search process, and The second is to define a requirement for early trimming of a significant number of non-closed candidates,to increase the performance of the proposed system by reducing the running time and memory requirements algorithm (LAM et al., 2021). Algorithm bellow discuss how CloGraMi based Frequent Sub-graph mining work. Algorithm above discovered by LAM[18] and published at IEEE access the main goal for this algorithm is to reducing the running time of calculating all frequent sub-graph and also for reducing memory requirements of frequent sub-graph storage. Algorithm 4 shown CloGraMi.



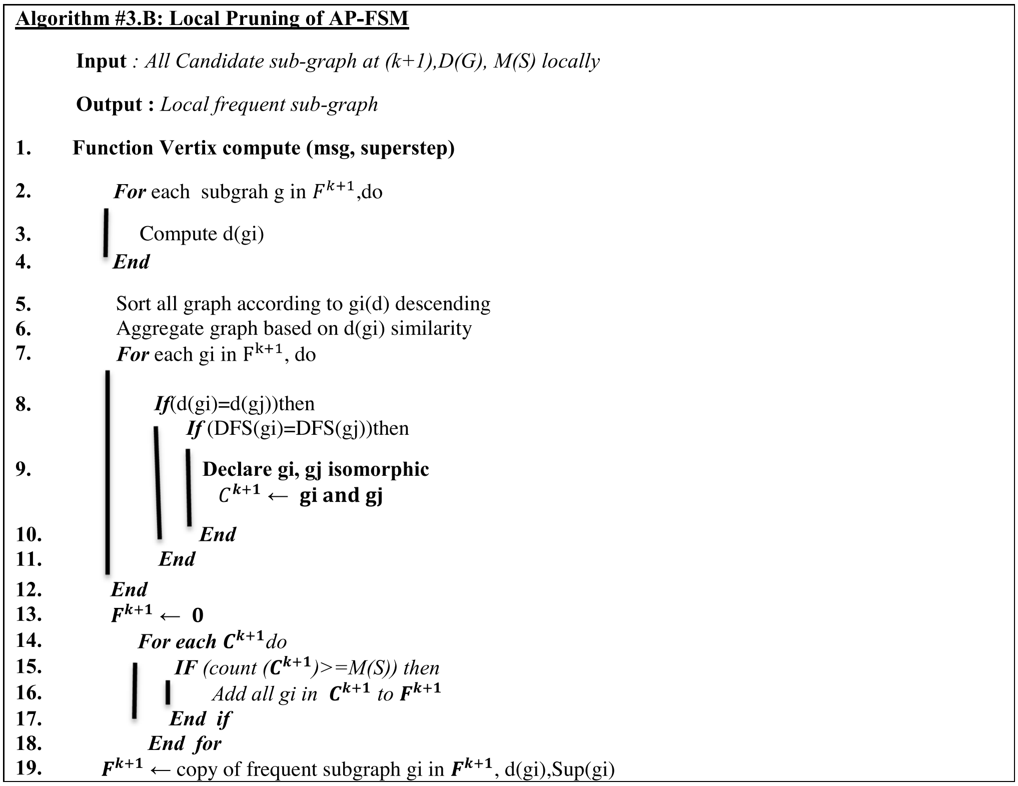
Algorithm #3: AP-FSM .*(Vandana Bhatia, Rinkle Rani 2018).*

```
Input:     G                                              // G:Sample graph.
Output:    F(S)                                           // F(S):  frequent Sub-graph
While (k<z)do
    // Procedure vertices compute (msg, superstep)
    IF (super step=0) then
        F=0
        F1← 1-FIS;                        // 1-FIS :one frequent itemset
        S(F1) ← All embedding of F1in G
        Else
        S($F^k$) ← All embedding of $F^k$ in G
        Ext ($F^k$) ← possible extension of all embedding of $F^k$ in G
        $F^{k+1}$ ← $F^{k+1}$ U Ext($F^k$, $G$, S($F^k$))
        Call procedure Local pruning ((I($F^{k+1}$)),d(gi),M(S))
    End if

    $F^{k+1}$ ← all frequent Sub-graph from workers
    Function Master compute (msg, superstep)
    IF (I($F^{k+1}$) ≠ 0)
        Call Global pruning (I($F^{k+1}$)),d(gi),M(S))          //Frequent Sub-graph
    End if
End
```



Algorithm #3.A: Extension AP-FSM

```
$F^{k+1}$=0
    For each embedding subgrah s in S(F)do
        For each $C^{k+1}$ belong to subgrah set $C^{k+1}$ do
            If ($F^k$) is the parent of $C^{k+1}$ then
                For each possible extension ext(e,v) or € in Ext(g) do
                    If ((g,e)   v then
                        If (v is frequent in Ext(s))then
                                $C^{k+1}$ ← $C^{k+1}$ +ext(e,v)
                        End
                        Else
                    $C^{k+1}$ ← $C^{k+1}$ +ext(e)
                    End
                End
            End
        End
    End
$F^{k+1}$ ← all frequent Size k+2
```

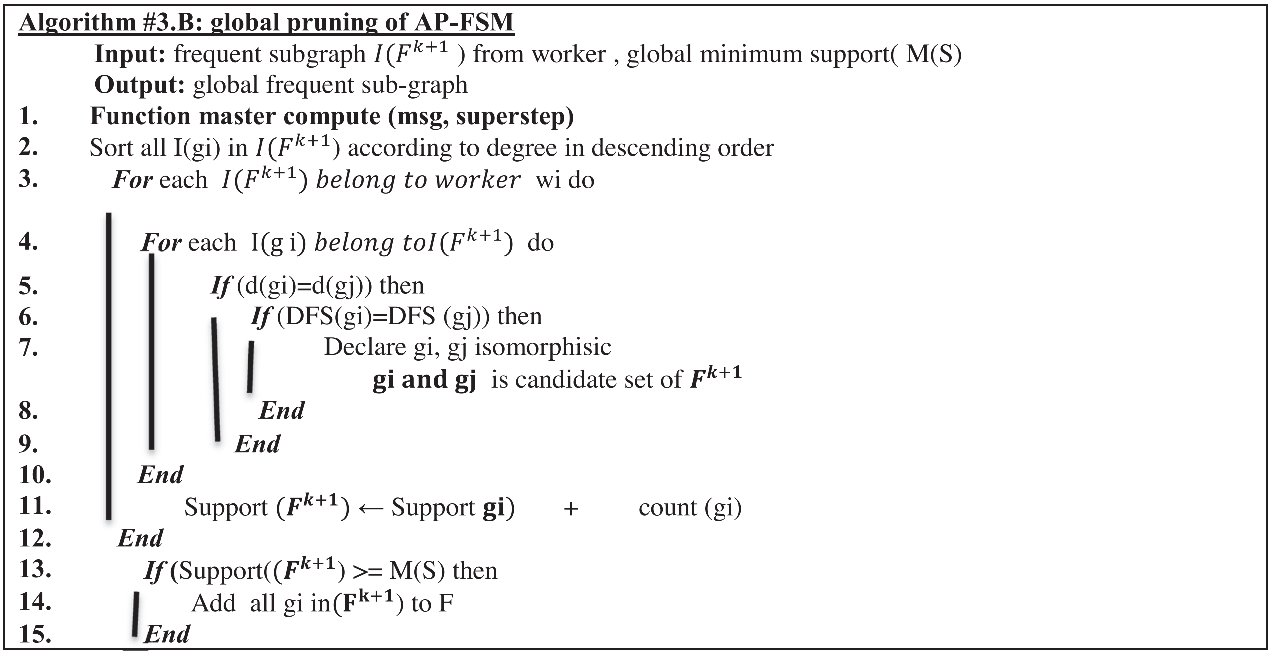


Algorithm #3.B: Local Pruning of AP-FSM

Input : *All Candidate sub-graph at (k+1),D(G), M(S) locally*

Output : *Local frequent sub-graph*

```
Function Vertix compute (msg, superstep)
    For each  subgrah g in $F^{k+1}$,do
        Compute d(gi)
    End
    Sort all graph according to gi(d) descending
    Aggregate graph based on d(gi) similarity
    For each gi in $F^{k+1}$, do
        If(d(gi)=d(gj))then
            If (DFS(gi)=DFS(gj))then
                Declare gi, gj isomorphic
                $C^{k+1} \leftarrow$ gi and gj
            End
        End
    End
    $F^{k+1} \leftarrow 0$
    For each $C^{k+1}$do
        IF (count ($C^{k+1}$)>=M(S)) then
            Add all gi in  $C^{k+1}$ to $F^{k+1}$
        End  if
    End  for
    $F^{k+1} \leftarrow$ copy of frequent subgraph gi in $F^{k+1}$, d(gi),Sup(gi)
```



Algorithm #3.B: global pruning of AP-FSM

Input: frequent subgraph $I(F^{k+1})$ from worker , global minimum support(M(S)

Output: global frequent sub-graph

```
Function master compute (msg, superstep)
Sort all I(gi) in $I(F^{k+1})$ according to degree in descending order
    For each  $I(F^{k+1})$ belong to worker  wi do
        For each  I(g i) belong toI($F^{k+1}$)  do
            If (d(gi)=d(gj)) then
                If (DFS(gi)=DFS (gj)) then
                    Declare gi, gj isomorphisic
                        gi and gj  is candidate set of $F^{k+1}$
                End
            End
        End
        Support ($F^{k+1}$) ← Support gi)      +       count (gi)
    End
    If (Support(($F^{k+1}$) >= M(S) then
        Add  all gi in($F^{k+1}$) to F
    End
```

Algorithm #4: CloGraMi *(Lam et. al, 2021)*

```
Input:          G, M(S)                         // G:data set of fix graph. M(S): minimum
                                                supported threshold.
Output:         F List                          //F List closed sub-graphs of G
Initialization  F List ← Φ

For each e ∈ F Edges do                         // F Edges is a set contains frequent edges of
                                                G
    F List ← F List ∪ Extend-By-Level(e, M(S) , F Edges,
    G)
    Remove e from F Edges
End
procedure Extend-By-Level (e, M(S), F Edge, G)
    For each g ∈g Sub-graphs do                 // g Sub-graphs: generated sub-graph
        if sG(g) ≥ M(S) then                    //if sG(g)
            F Sub-graphs ← F Sub-graphs ∪g          F Sub-graphs is frequent sub-
                                                graph
            if sG(g) = sG(S) then               //compare the support
                closed ← false
            Else closed ← True
            End
        End
    C Sub-graphs ← C Sub-graphs ∪S              //C Sub-graphs: list of closed sub-graphs
    End
```

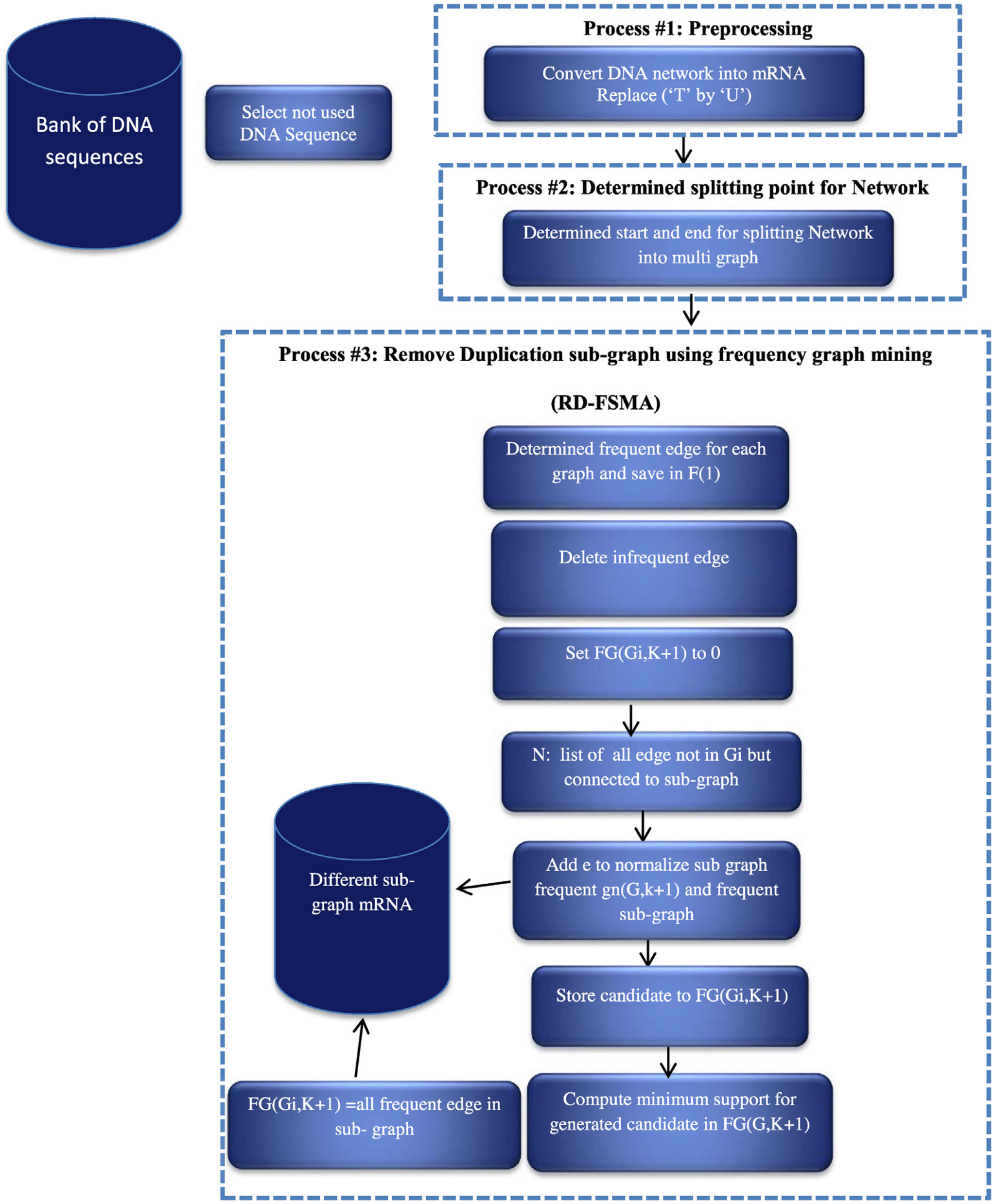
Algorithm #5: Hybrid-Tree-Miner .(Yun Chi et, al., 2004).

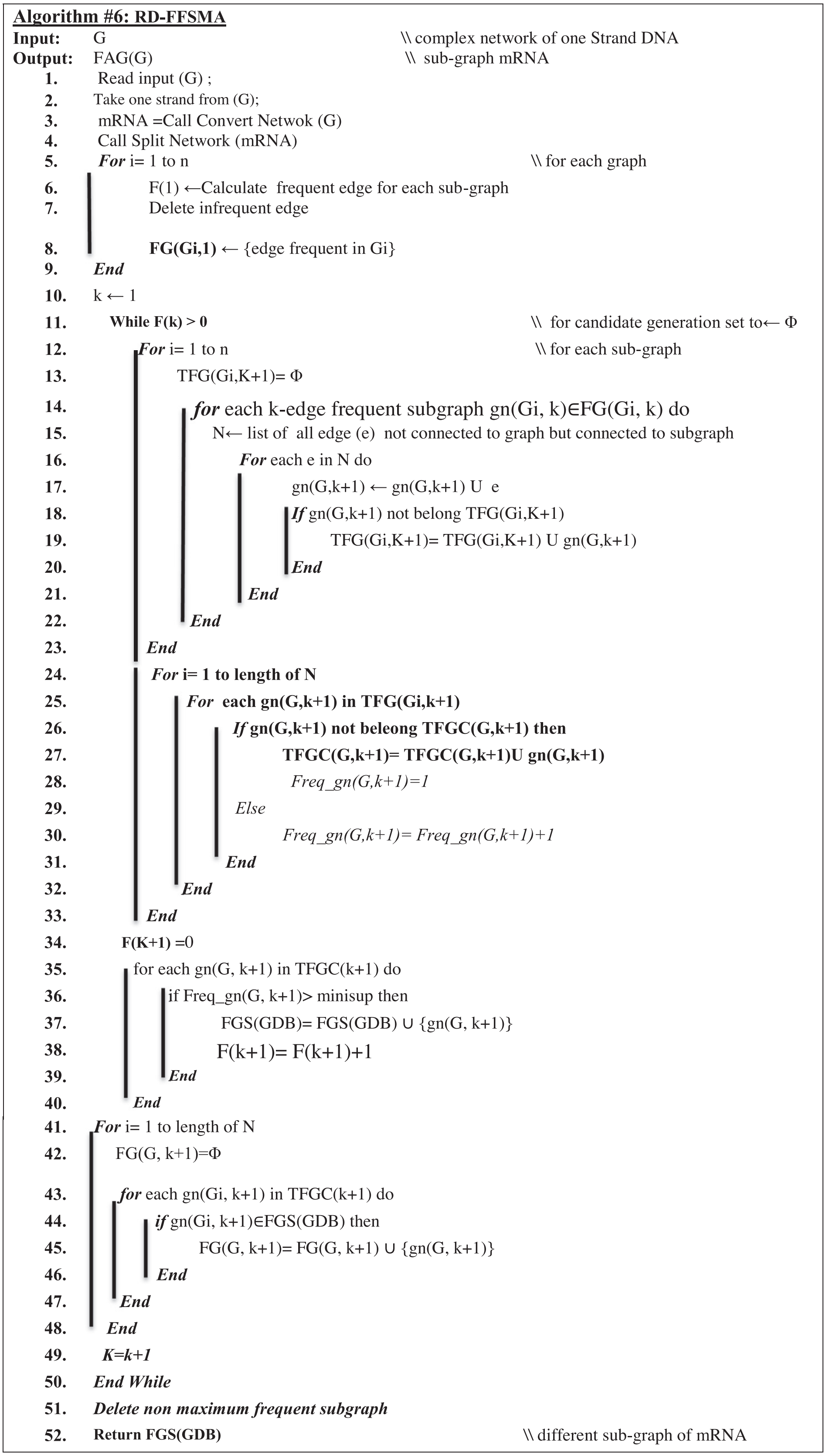
```
Input:      F1tree,F2tree , M(S)        //  F1tree,F2tree: frequent one and two respectively
Output:     F(S)                        // F(S):        frequent Sub-graph
    F(S)← F1tree U F2tree
    C← sort (F2tree)
    Call procedure Grow(C,F(S),M(S)
    End
Procedure Grow(C,F(S),M(S)
    For i ← 1: C do
        For j ← i: C do
            P← join (ci,ci+1)
          IF supp(p) ≥ minsup
              then J ← J ∪ p;
              F ← F ∪ J;
          End
          Enum-Grow(J,F,minsup);
        End
    End
    E ← ∅;
    For each leg lm of ci do
        For each possible new leg ln do
            q ← ci plus leg ln at position lm;
            IF  supp(q) ≥ minsup
                E ← E ∪ q
            End
            F ← F ∪ E;
        End
    Grow(E,F,minsup);F ← FUP
    End
```

Where: G is a graph database, d(g) is degree of each sub-graph,F(S) is frequency of all sub-graph, MF(S) is a maximum frequent sub-graph extracted, I and j is iteration of algorithm, C code is a candidate code in form of binary, Gn is a normalize incidence matrix, close is a close frequent sub-graph.

4. A Novel algorithm to enhancement the performance and reduce the computations (RF-FFGM)

Fast Frequent sub-graph mining (FSM) is an important part of graph mining (Jia Wu1, 2008). It has gotten a lot of attention in fields including web data mining, social networks and bioinformatics. Within the topic of data mining, graph mining has become a well-established discipline. It has sparked a lot of interest in the previous decade, thanks to developments in computer hardware that have enabled large-scale graph data mining to be done. [21,1] The task of mining common sub-graphs from graph databases is a simple one with a wide range of applications. Finding all sub-graphs that appear more than a certain threshold value is known as frequent sub-graph mining. Candidate creation and frequency calculation are the two basic phases. Most existing work in the candidate generation stage starts with a frequent edge or vertex to generate frequent candidate patterns. Kavitha[14] The extract useful knowledge from complex network of DNA is a complex phenomenon which plays an important role in clinical, treatment. For this problem, mining of frequent sub-structure of DNA is important issue for extracting meaningful knowledge from DNA. However, using this method may increase the computing time of calculating such important knowledge. [29] In this, work, we extract all frequent sub-structure in large complex DNA by using RD-FSMA algorithm. Algorithm begin by entering complex large network of DNA from bank of DNA available online. This algorithm divided into three main Step, first step is preprocessing for preparing network for using in farther stage. In this stage large complex network of DNA is transferring into mRNA structure to extract meaningful knowledge from it. Second step is splitting network of mRNA into multiple Sub-graph of mRNA. Third step multiple sub-graph enter to RD-FSM algorithm that contained multiple stage to calculate and save different sub-graph extracted from RD-FSMA algorithm in buffer to reduce the computation time for calculated all frequent sub-structure mRNA. Figure bellow illustrate the main step of algorithm.


Bank of DNA sequences
Select not used DNA Sequence
Process #1: Preprocessing
Convert DNA network into mRNA Replace ('T' by 'U')
Process #2: Determined splitting point for Network
Determined start and end for splitting Network into multi graph
Process #3: Remove Duplication sub-graph using frequency graph mining
(RD-FSMA)
Determined frequent edge for each graph and save in F(1)
Delete infrequent edge
Set FG(Gi,K+1) to 0
N: list of all edge not in Gi but connected to sub-graph
Add e to normalize sub graph frequent gn(G,k+1) and frequent sub-graph
Different sub-graph mRNA
Store candidate to FG(Gi,K+1)
Compute minimum support for generated candidate in FG(G,K+1)
FG(Gi,K+1) =all frequent edge in sub- graph



Algorithm #6: RD-FFSMA

```
Input:  G                                         \\ complex network of one Strand DNA
Output: FAG(G)                                    \\ sub-graph mRNA
 Read input (G) ;
 Take one strand from (G);
 mRNA =Call Convert Netwok (G)
 Call Split Network (mRNA)
 For i= 1 to n                                    \\ for each graph
     F(1) ←Calculate  frequent edge for each sub-graph
     Delete infrequent edge
     FG(Gi,1) ← {edge frequent in Gi}
 End
 k ← 1
 While F(k) > 0                                   \\  for candidate generation set to← Φ
     For i= 1 to n                                \\ for each sub-graph
         TFG(Gi,K+1)= Φ
         for each k-edge frequent subgraph gn(Gi, k)∈FG(Gi, k) do
             N← list of  all edge (e)  not connected to graph but connected to subgraph
             For each e in N do
                 gn(G,k+1) ← gn(G,k+1) U  e
                 If gn(G,k+1) not belong TFG(Gi,K+1)
                     TFG(Gi,K+1)= TFG(Gi,K+1) U gn(G,k+1)
                 End
             End
         End
     End
     For i= 1 to length of N
         For  each gn(G,k+1) in TFG(Gi,k+1)
             If gn(G,k+1) not beleong TFGC(G,k+1) then
                 TFGC(G,k+1)= TFGC(G,k+1)U gn(G,k+1)
                 Freq_gn(G,k+1)=1
             Else
                 Freq_gn(G,k+1)= Freq_gn(G,k+1)+1
             End
         End
     End
     F(K+1) =0
     for each gn(G, k+1) in TFGC(k+1) do
         if Freq_gn(G, k+1)> minisup then
             FGS(GDB)= FGS(GDB) ∪ {gn(G, k+1)}
             F(k+1)= F(k+1)+1
         End
     End
 For i= 1 to length of N
     FG(G, k+1)=Φ
     for each gn(Gi, k+1) in TFGC(k+1) do
         if gn(Gi, k+1)∈FGS(GDB) then
             FG(G, k+1)= FG(G, k+1) ∪ {gn(G, k+1)}
         End
     End
 End
 K=k+1
 End While
 Delete non maximum frequent subgraph
 Return FGS(GDB)                                  \\ different sub-graph of mRNA
```

```
Algorithm #6.1: Convert Network

Input:   G                                \\ complex network of one Strand DNA
Output:  mRNA                             \\ mRNA network
   Read input (G)
   For i = 0 to length of (G)
         If G(i) = =("T")
               mRNA(i)= ("U")             \\ replace Thymine by Uracil
         Else
         mRNA(i)=G(i)
         End
   End
```

```
Algorithm #6.2: Split Network

Input:   mRNA                             \\ complex network of one Strand DNA
Output:  Set of Graph (G1,G2,.....Gn)     \\ mRNA network
   Start ←1
   Read input (mRNA)
   for i = 0 to length of (mRNA)
         End= (Round(length of mRNA)/length of mRNA-Start)*3+Start

         Start=End+1
         for i = Start to End
               G(i)=mRNA(i)
               If End> length(mRNA)
                     End=length(mRNA)
               End
         End
   End
```

5. Implementation of A novel algorithm (RD-FFGM)

Given a Graph Data set represent (graph of double helix DNA Structure) that continue huge number of four element,(A, T, C, G) connected from one side of DNA strand to another side. Connection element is not arbitrary it done in legal form: (A with T) and (C with G). These elements connected in one strand by three components (phosphate, sugar and base itself). Frist algorithm begin by take one strand of DNA by separate it from double helix and work with it. Then convert these structures of DNA into Messenger RNA (mRNA) this step can be done by replacing every Thymine (T) element by Uracil (U) element and stay Cytosine (C) and Guanine (G) without change for all structures of mRNA. After this step all graph (mRNA Strand) must cut into multiple sub-graph dataset then enter to RD-FSM algorithm to find different Sub-graph possible for given minimum supported threshold. Example bellow can show how process will done.

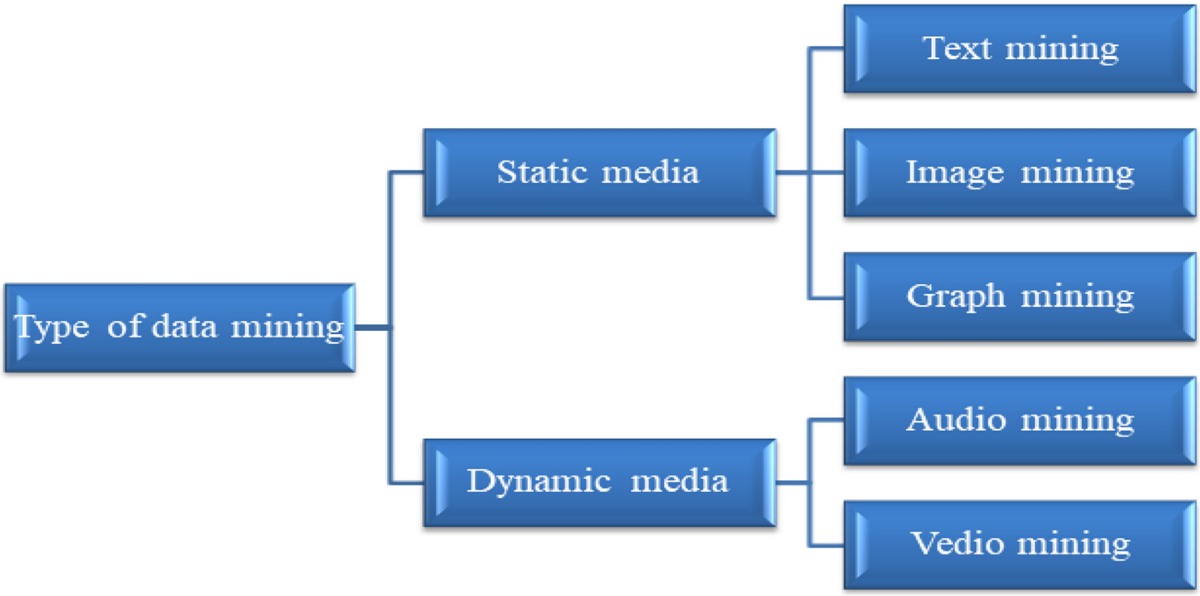


Fig. 1. General data mining area used (Perner, 2002) (ManjunathS. Balaji, 2014).

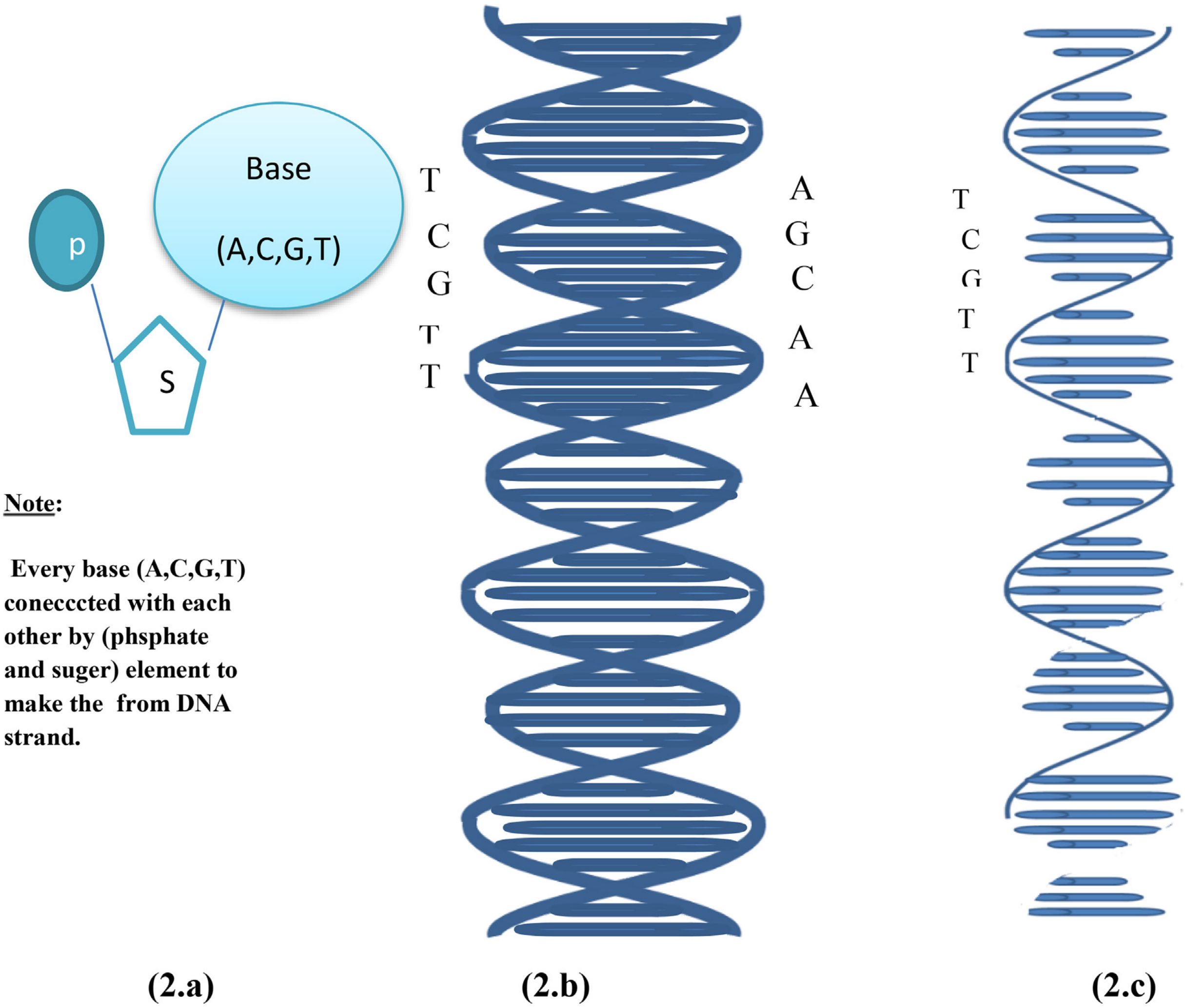

Base
(A,C,G,T)
p
S
Note:
Every base (A,C,G,T) coneccted with each other by (phsphate and suger) element to make the from DNA strand.
T
C
G
T
T
A
G
C
A
A
T
C
G
T
T
(2.a)
(2.b)
(2.c)

Fig. 2. DNA form (2.a) specification of one element connection in one strand (2.b) Double helix DNA connection one side to another in legal form (2.c) one strand DNA.

Table 1
The main concepts of researcher related to literature survey.

Author(s)	Data set/ Database	Preprocessing	Methodology	Evaluation tools	Advantage	Disadvantage
Lopamudra [16]	■ RNA Sequence ■ SARS-CoV-2-human PPI database ■ Negative dataset ■ independent dataset	LVQ for feature subset selection	(SVM), (NB), (RF), (KNN) with multilayer perceptron	■ confusion matrix	■ predict Protein -Protein Interaction between human protein and viruses ■ discover anti-COVID drug ■ various length of Protein human sequences	■ Need biological knowledge
Minghui Wang [17]	independent test set (protein sequence) https://github.com/QUST-AIBBDRC/SulSite-GTB/.	Feature encoding	SulSite-GTB	■ Accuracy	■ identification of protein S-sulfenylation high accuracy of prediction	■ Time complexity
Xiujuan Zhao et al. [27]	RNA Sequence)https://github.com/zhaoxj-tech/DFpin.git.(	remove feature redundancy	■ DF pin	■ Accuracy	■ predicting protein interaction related to different type of human Disease	■ Time complexity
Zicheng [17]	RNA SequenceMSKCC dataset (https:// https://www.mskcc.org/)TCGAPRAD dataset (https://portal.gdc.cancer.gov/)	Segmentation	A novel immune–related RNA–binding proteins signature	■ Pearson Correlation analysis	■ identify immune-related Retinol-binding proteins (RBP) are a family of proteins with diverse functions ■ predict prognosis and therapy response in prostate cancer	■ Time complexity
Chang Woo el al 2022	RNA sequence (https://www.uniprot.org/downloads)	Segmentation od RNA	F UTUSA	■ Accuracy	■ predict protein function based on information of sequence of RNA only	Large computation

Table 2
Comparison between famous five type of graph mining.

T	Advantage	Disadvantage	Main parameter	Secondary parameter
GSpan (Yan,2002)	■ GSpan algorithm have high accuracy for generate all frequent sub-graph mining for a given graph dataset. ■ Find all frequent sub-graphs without all candidate generation, and false positives pruning. ■ Efficient algorithm for reducing duplicate, graph generation. ■ Work on labelled graph. ■ Less memory consuming.	■ Large computation because it used Depth first search algorithm DFS algorithm and backward search for verification. ■ Take large time complexity. ■ Used for small graph. ■ Problem with unlabelled graph. ■ Work on static graph	■ G: labelled graph dataset. ■ S: code of depth first search.	■ M(s):minimum supported Threshold
FSM *(Huan, 2003)*	■ FFSM algorithm have good accuracy for generate all frequent sub-graph mining for a given graph dataset ■ Take fast time for calculate all frequency sub-graph, because scanning all at once. Work on labelled graph. ■ Vertical search technique that limit number of candidate	■ FFSM algorithm have Large Memory requirement because of (canonical adjacency matrix (CAM)) for both frequency edge and vertex. ■ Problem with unlabelled graph. ■ Work on static graph	■ G: labelled graph datasetM (s) :minimum supported Threshold desired for choosing frequent sub-graph mining ■ CAM, matrixes (Pa,Sa)	■ M(s):minimum supported Threshold ■ N: Graph size
Hybrid-Tree-Miner (Yun Chi et, al., 2004).	■ Work on huge dataset ■ Work on Large tree works on both rooted and un rooted trees	■ Time Complexity because used breadth – first search in enumeration tree stage ■ Work on static graph	■ F1tree,F2tree: frequent one and two respectively ■ M(S): minimum supported Threshold desired for choosing frequent sub-graph mining	■ M(s):minimum supported Threshold ■ N: Graph size
AP-FSM (Vandana Bhatia and Rinkle Rani, 2018)	■ AP-FSM algorithm have good accuracy for generate all frequent sub-graph mining for a given graph dataset. ■ Take one large sample graph for a given many graph. ■ Work on distributed environment	■ Take only one large sample graph for a given many graph. ■ Work only on parallel environment ■ Work on static graph	■ G: labelled graph dataset.	■ M(s):minimum supported Threshold
CloGraMi (LAM et, al 2021)	■ Work on huge dataset ■ Work on Large graph Work on direct and underact graph ■ Less memory require	■ Time Complexity ■ Work on static graph	■ G: labelled graph dataset.	■ M(s):minimum supported Threshold

Table 3
Comparison in the Main Parameters, which affect mining sub-graph techniques.

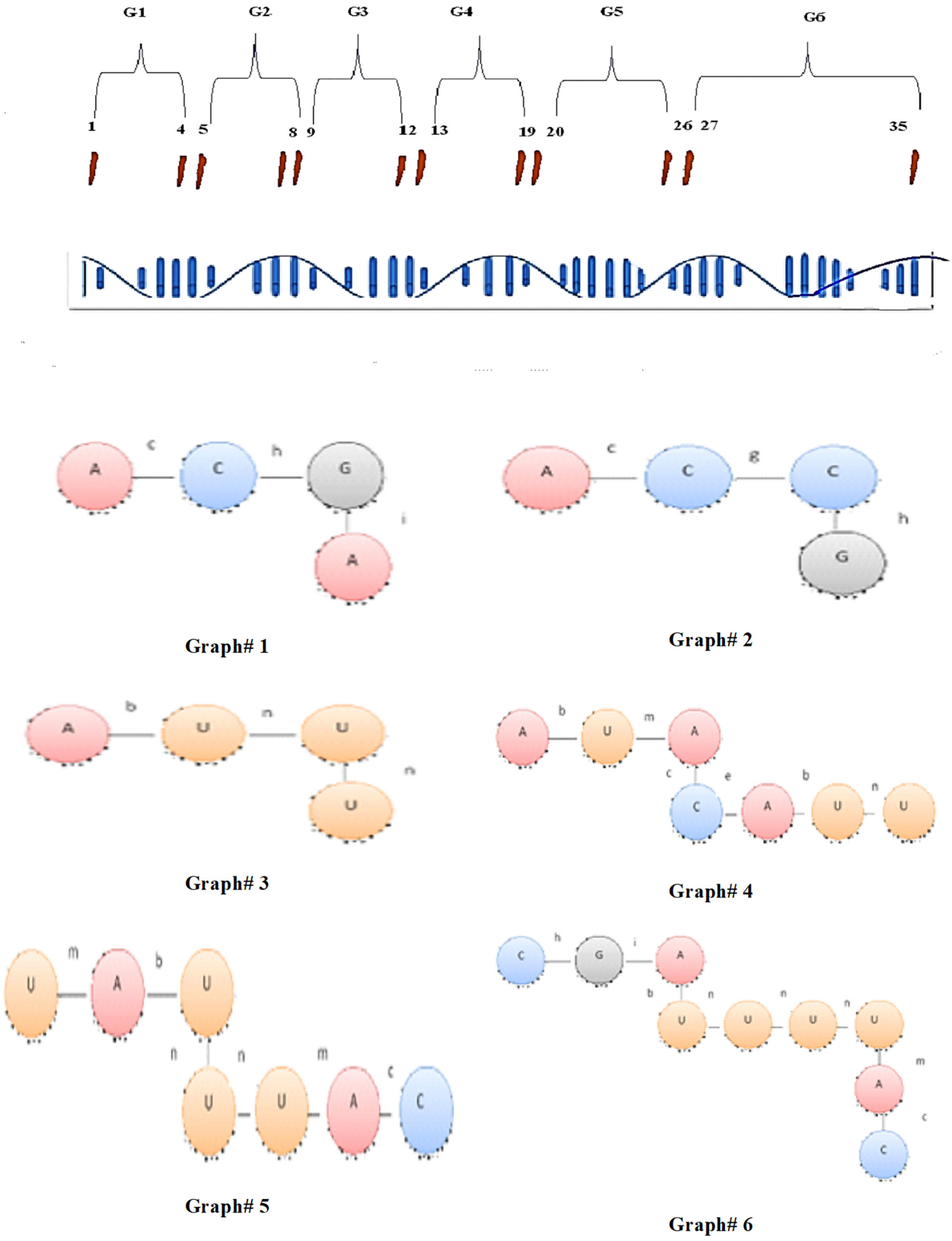
Algorithm/ parameter	G	d(g)	M(S)	F(S)	MF(S)	i	j	C code	Gn	close
Gspan		×	✓	✓	×	✓	×	×	×	×
Fsm	✓	×	✓	✓	×	✓	×	×	×	×
Hybrid tree miner		×	✓	✓	×	✓	×	×	×	×
Ap-FSM	✓	✓	✓	✓	×	✓	×	×	×	×
CoGrami	✓	×	✓	×	×	✓	×	×	×	✓
FFSM	✓	×	✓	✓	✓	✓	✓	✓	✓	✓

Table 4
Splitting whole network of mRNA extracted into multi-graph.

Graph	Start	End
G1	1	4
G2	5	8
G3	9	12
G4	13	19
G5	20	26
G6	27	35

Table 5
Splitting whole network of mRNA extracted into multi-graph.

Graph	Sequence
G1	ACGA
G2	ACCG
G3	AUUU
G4	AUACAUU
G5	UAUUUAC
G6	CGAUUUUAC


G1
G2
G3
G4
G5
G6
1
4
5
8
9
12
13
19
20
26
27
35
A c C h G i A
A c C g C h G
A b U n U n U
A b U m A c C e A b U n U
U m A b U n U n U m A c C
C h G i A b U n U n U n U m A c C

Graph# 1

Graph# 2

Graph# 3

Graph# 4

Graph# 5

Graph# 6

Fig. 3. Splitting whole network of mRNA extracted into multi-graph.

5.1. Complex network of DNA

Step 1: for a given complex Network of DNA, take one strand of it and start work on it.

Step 2: convert one strand of taken DNA into mRNA by replace every "Thymine" base into "Uracil" base and keep both "Cytosine" and "Guanine" without change.

Step 3: Split whole mRNA network into multiple sub-graph G1, G2,.. G n. this process cannot be done randomly therefore, we calculate start and end point to segment network into multiple graphs in legal form, according to start and end point calculated. Suppose first start point begin from begin of Network and end point according to equation bellow from given length of mRNA and start point so the start an end point illustrate bellow.

Suppose sequence of mRNA is:

ACGAACCGAUUUAUACAUUUAUUUACCGAUUUUAC.

End = Round (Len(mRNA)/Len(mRNA)-Start))*3 + Start. Figs. 1 and 2. Tables 1-5.

Our first stage of work of segmented network of mRNA illustrates in Fig. 3 bellow:

Then for each graph extracted we proposed a label on each base connection according to four elements of one connection, illustrated in Table 6:

Step 4: For all graph in database of network mRNA must find frequent for one edge and save it into F(1) and must compute for each Graph extracted from network Incidence matrix normalization (Iia Wu, 2008) to compute all frequent sub-graph from all graph from Incidence matrix normalization, this process represented bellow:

All other edge not found in graph ignore, {a, d, j, k, l, o, p}. Now for every graph extracted we build type connection between base in form of row column matrix, if edge connected between nodes represent 1, otherwise 0. And build incidence matrix normalization for every graph according to (Jia Wu et al., 2008). row at the first time and then column ordered descending according to coding content and represented bellow for each graph. Tables 7-23.

Step 5: Now for each frequent one edge calculated, find frequent sub-graph of (K + 1). In each graph collect set of **N** edge not belong to graph itself but connected to sub-graph frequent in its node in other graph.

Then store each unique candidate from each graph in (TFGC edge) for use it in next (K + 1) frequent sub-graph generation.

Table 6
Connection label on connected base.

#	Nods	Edge
	AA	a
	AU	b
	AC	c
	AG	d
	CA	e
	CU	f
	CC	g
	CG	h
	GA	i
	GU	j
	GC	k
	GG	l
	UA	m
	UU	n
	UC	o
	UG	p

Table 7
Frequent one edge.

#	Edge	F(1) edge
1	(A, c, C,)	5
2	(C, h, G)	3
3	(G, I, A)	2
4	(A, b, U)	5
5	(U, n, U)	8
6	(U, m, A)	4
7	(C, e, A)	1
8	(C, g, C)	1

Table 8
(8.a) matrix code binary connection of graph #1. (8.b) normalize incidence in form of row. (8.c) normalize incidence in form of column.

G1	c	h	i
A	1	0	0
C	1	1	0
G	0	1	1
A	0	0	1
Gn1(row)	c	h	i
C	1	1	0
A	1	0	0
G	0	1	1
A	0	0	1
Gn1(row,colomn)	c	h	i
C	1	1	0
A	1	0	0
G	0	1	1
A	0	0	1

Table 9
(9.a) matrix code binary connection of graph# 2. (9.b) normalize incidence in form of row. (9.c) normalize incidence in form of column.

G2	c	g	h
A	1	0	0
C	1	1	0
C	0	1	1
G	0	0	1
G2(row)	c	g	h
C	1	1	0
A	1	0	0
C	0	1	1
G	0	0	1
G2(row,column)	c	g	h
C	1	1	0
A	1	0	0
C	0	1	1
G	0	0	1

Therefore for each sub-graph extracted in each incidence normalize add another edge from graph to each sub graph extracted a according to adjacent edge in each frequent two edge calculated, at this way compute all frequent sub-graph so.

Take one different four edge and store it in buffer for use it.

Table 10
(10.a) matrix code binary connection of graph 3. (10.b) normalize incidence in form of row. (10.c) normalize incidence in form of column.

G3	b	n	n
A	1	0	0
U	1	1	0
U	0	1	1
U	0	0	1

G3(row)	b	n	n
U	1	1	0
A	1	0	0
U	0	1	1
U	0	0	1

G3(row,column)	b	n	n
U	1	1	0
A	1	0	0
U	0	1	1
U	0	0	1

Table 11
(11.a) matrix code binary connection of graph# 4. (11.b) normalize incidence in form of row. (11.c) normalize incidence in form of column.

G4	b	m	c	e	b	n
A	1	0	0	0	0	0
U	1	1	0	0	0	0
A	0	1	1	0	0	0
C	0	0	1	1	0	0
A	0	0	0	1	1	0
U	0	0	0	0	1	1
U	0	0	0	0	0	1

Gn4(row)	b	m	c	e	b	n
U	1	1	0	0	0	0
A	1	0	0	0	0	0
A	0	1	1	0	0	0
C	0	0	1	1	0	0
A	0	0	0	1	1	0
U	0	0	0	0	1	1
U	0	0	0	0	0	1

Gn4(row, column)	b	m	c	e	b	n
U	1	1	0	0	0	0
A	1	0	0	0	0	0
A	0	1	1	0	0	0
C	0	0	1	1	0	0
A	0	0	0	1	1	0
U	0	0	0	0	1	1
U	0	0	0	0	0	1

Table 12
(12.a) matrix code binary connection of graph# 5. (12.b) normalize incidence in form of row. (12.c) normalize incidence in form of column.

Gn5 (row)	m	b	n	n	m	c
A	1	1	0	0	0	0
U	1	0	0	0	0	0
U	0	1	1	0	0	0
U	0	0	1	1	0	0
U	0	0	0	1	1	0
A	0	0	0	0	1	1
C	0	0	0	0	0	1

Gn5 (row, coumn)	m	b	n	n	m	c
A	1	1	0	0	0	0
U	1	0	0	0	0	0
U	0	1	1	0	0	0
U	0	0	1	1	0	0
U	0	0	0	1	1	0
A	0	0	0	0	1	1
C	0	0	0	0	0	1

Table 13
(13.a) matrix code binary connection of graph #6. (13.b) normalize incidence in form of row. (13.c) normalize incidence in form of column.

G6	h	i	b	n	n	n	m	c
C	1	0	0	0	0	0	0	0
G	1	1	0	0	0	0	0	0
A	0	1	1	0	0	0	0	0
U	0	0	1	1	0	0	0	0
U	0	0	0	1	1	0	0	0
U	0	0	0	0	1	1	0	0
U	0	0	0	0	0	1	1	1
A	0	0	0	0	0	1	1	1
C	0	0	0	0	0	0	0	1

Gn6(row)	h	i	b	n	n	n	m	c
G	1	1	0	0	0	0	0	0
C	1	0	0	0	0	0	0	0
A	0	1	1	0	0	0	0	0
U	0	0	1	1	0	0	0	0
U	0	0	0	1	1	0	0	0
U	0	0	0	0	1	1	0	0
U	0	0	0	0	0	1	1	1
A	0	0	0	0	0	1	1	1
C	0	0	0	0	0	0	0	1

Gn6(row, column)	h	i	b	n	n	n	m	c
G	1	1	0	0	0	0	0	0
C	1	0	0	0	0	0	0	0
A	0	1	1	0	0	0	0	0
U	0	0	1	1	0	0	0	0
U	0	0	0	1	1	0	0	0
U	0	0	0	0	1	1	0	0
U	0	0	0	0	0	1	1	1
A	0	0	0	0	0	1	1	1
C	0	0	0	0	0	0	0	1

Table 14
(14.a) frequent one edge from normalize incidence matrix and set of n edge not in graph but connected with frequent edge in this graph in other graph. (14.b) frequent k + 1.

Gn1 frequent K	F (1) Edge	F (1) edge
(A, c, C)	(A, c, C)	5
(C, h, G)	(C,h,G)	3
(G, i, A)	(G, i, A)	2
N(set of edge not belong to G1 but connected with edge of sub graph above in other graph,so N= {c,(g, e, m), h,(g), I,(0)}		

Gn1 frequent K+1	F (2) edge
(A, c, C, g, C)	1
(A,c, C,e,A)	1
(U, m, A, c, C)	3
(C, g,C,h, G)	1

Table 15
(15.a) frequent one edge from normalize incidence matrix and set of n edge not in graph but connected with frequent edge in this graph in other graph. (15.b) frequent k + 1.

Gn2 frequent K + 1	F(1) Edge	F(1) edge
(A, c, C)	(A, c, C)	5
(C, g, C)	(C, g, C)	1
(C, h, G)	(C, h, G)	3
N(set of edge not belong to c, g, h) but connected with edge of sub graph above in other graph,so N= {c=(e, m), g=(0),h=(i)		

Gn2 frequent K+1	F(2) edge
(A, c, C, e, A)	1
(U, m, A, c, C)	3
(C, h, G, i, A)	2

Table 16
(16.a) frequent one edge from normalize incidence matrix and set of n edge not in graph but connected with frequent edge in this graph in other graph. (16.b) frequent k + 1.

Gn3 frequent	F(1) Edge	F(1) edge
(A, b, U)	A b U	5
(U, n, U)	U n U	8
N(set of edge not belong to c, h, i) but connected with edge of sub graph above in other graph,so N= {b=(i, m), n=(m).		
Gn3 frequent K+1		F(2) edge
(G, i, A, b, U)		1
(U, n, U, m, A)		2

Table 17
(17.a) frequent one edge from normalize incidence matrix and set of n edge not in graph but connected with frequent edge in this graph in other graph. (17.b) frequent k + 1.

Gn4 frequent	F(1) Edge	F(1) edge
(A, b, U)	A b U	2
(U, m, A)	U m A	4
(A, c, C)	A c C	5
(C, e, A)	C e A	1
(A, b, U)	A b U	5
(U, n, U)	U n U	8
N(set of edge not belong to b, m, c, e, n) but connected with edge of sub graph above in other graph,so N= {b=(i), n=(0) c= (g, h). e = 0, m=(0)		
Gn4 frequent K+1		F(2) edge
(G, i, A, b, U)		1
(A, c, C, g,C)		1
(A, c, C, h,G)		1

Table 18
(18.a) frequent one edge from normalizes incidence matrix and set of n edge not in graph but connected with frequent edge in this graph in another graph. (18.b) frequent k + 1.

Gn5 frequent	F(1) Edge	F(1) edge
(U, m, A)	U, m, A	4
(A, b, U)	A, b, U	5
(U,n, U)	U, n, U	8
(A,c, C)	A c C	5
N(set of edge not belong to b, m, c, n) but connected with edge of sub graph above in other graph,so N= {m=(0), b=(i) n= (0). c=(e,h,g)		
Gn5 frequent K+1		F(2) edge
(G, i, A, b, U)		1
(A, c, C, e,A)		1
(A, c, C, g,C)		1
(A, c, C, h,G)		1

Table 19
(19.a) frequent one edge from normalizes incidence matrix and set of n edge not in graph but connected with frequent edge in this graph in another graph. (19.b) frequent k + 1.

Gn6 frequent	F(1) Edge	F(1) edge
(C, h, G)	A c C	5
(G, i, A)	C h G	3
(A, b, U)	G i A	2
(U, n, U)		
(U, m, A)		
(A,c,C)		
N(set of edge not belong to G6 but connected with edge of sub graph above in other graph,so N= {h=(g), i=(0), b=(0), n = 0, m = 0, c = g		
Gn6 frequent K+1		F(2) edge
(C, g, C, h,G)		1
(A, c, C, g,C)		1

Table 20
Frequent of two edges.

#	Edge	TFGC edge
	(U, m, A, c, C)	3
	(C, h, G, i, A)	2
	(G, i, A, b, U)	1
	(U, n, U, m, A)	2
	(A, c, C, g,C)	1
	(A, c, C, h,G)	1
	(A, c, C, e,A)	1
	(C, g, C, h,G)	1

Table 21
Frequent three edge.

#	Edge	FG(G, k + 1) + 1
	(U,n,U, m, A, c, C)	2
	(A,c,C, h, G, i, A)	1
	(C, h,G, i, A, b, U)	1
	(U,n,U, n, U, m, A)	2
	(U, n, U, m, A, c, C)	2
	(A, c, C, g,C, h, G)	1
	(A, c, C, h,G, I, A)	1
	(A, c, C, e,A,b,U)	1

Table 22
Frequent four edge.

#	Edge	FG(G, k + 1) + 1
	(U,n,U,n,U, m, A, c, C)	2
	(U,n,U,n,U, n, U, m, A)	1
	(A, c, C, e,A, b,U, n, U)	1

Table 23
Buffer of Different Frequent Edge.

#	Buffer	Number
	(U,n,U,n,U, m, A, c, C)	1
	(U,n,U,n,U, n, U, m, A)	1
	(A, c, C, e,A, b,U, n, U)	1

6. Conclusions

In this work a new intelligent deep analysis algorithm called reduce frequency based on fast frequency graph mining (RF-FFGM) is established; also it shown deep analysis to graph mining techniques ***(GSpan, FFSM, Hybrid-Tree-Miner, Approximate Frequent Sub-graph, CloGraMi and FFSM) this analysis focus on determined (the main programming steps, main parameters, advantages, disadvantages) for each algorithm.*** Finally determine the association rules for the sub-qstructure after running the algorithm and identifying the final sub-structure [15]. An association rule's confidence is a percentage value that indicates how often the rule head appears in all of the groups that contain the rule body. This rule's confidence value reflects how trustworthy it is. If all body items are known to be contained in a group, the greater the value, the more probable the head items will exist in that group. JesminNahar[12] We discovery FFGM finds the frequent in a short time, because it building matrix code for connection edge and transforming matrices into incidence matrix, also; we found FFGM can get all the edges that have the highest contact with the other edges, so from the second stage therefore it avoids us from going through a sequential path to find duplicate edges. RF-FFGM appears as a pragmatic algorithm, it proves their robust to work with DNA sequence to reduce the computation and time.

Author contributions

All authors contributed to the study's conception and design. Design the system, test and analysis were performed by [Samaher Al-Janabi and Zena A. Kadhuim]. The first draft of the manuscript was written by [Samaher Al-Janabi]. All authors have read and agreed to the published version of the manuscript.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Al-Janabi S, Alkaim AF. A nifty collaborative analysis to predicting a novel tool (DRFLLS) for missing values estimation. Springer, Soft Comput 2020;24 (1):555–69. doi: https://doi.org/10.1007/s00500-019-03972-x.

[2] Bhatia V, Rani R. Ap-FSM: A parallel algorithm for approximate frequent subgraph mining using Pregel. Expert Syst Appl 2018;106:217–32.

[3] Bhatt CA, Kankanhalli MS. (2011) ''Multimedia data mining: state of the art and challenges. Multimed Tools Appl 2011;51:35–76. doi: https://doi.org/10.1007/s11042-010-0645-5.

[4] Ko CW, Huh J, Park J-W. Deep learning program to predict protein functions based on sequence information. MethodsX 2022;9:32. doi: https://doi.org/10.1016/j.mex.2022.101622.

[5] Cheng H., Yu J.X. (2018) ''Graph Mining. In: Liu L., Özsu M.T. (eds) Encyclopedia of Database Systems. Springer, New York, NY. https://doi.org/10.1007/978-1-4614-8265-9_80737.

[6] Chi, Yun & Yang, Yirong & Muntz, Richard. (2004). ''HybridTreeMiner: An Efficient Algorithm for Mining Frequent Rooted Trees and Free Trees Using Canonical Forms. 10.1109/SSDM.2004.1311189.

[7] S. H. Ali. ''A novel tool (FP-KC) for handle the three main dimensions reduction and association rule mining,'' IEEE, 2012 6th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), Sousse. 2012. 951-961. doi: 10.1109/SETIT.2012.6482042.

[8] Hand DJ. ''Principles of Data Mining. Drug-Safety 2007;30:621–2. doi: https://doi.org/10.2165/00002018-200730070-00010.

[9] Hsu W., Lee M.L., & Zhang J. (2002) ''Image Mining: Trends and Developments*: Journal of Intelligent Information Systems.* **19.** 7–23. https://doi.org/10.1023/A:1015508302797.

[10] Huan J., Wang W., & Prins J. (2003) ''Efficient mining of frequent subgraphs in the presence of isomorphism. In The proceedings of 3rd IEEE international conference on data mining (pp. 2–5). https://doi.org/10.1109/ICDM.2003.1250974.

[11] Al-Janabi S, Mahdi MA. Evaluation prediction techniques to achievement an optimal biomedical analysis. Int J Grid and Utility Computing 2019;10 (5):512–27.

[12] Nahar J, Imam T, Tickle KS, Chen Y-P. ''ssociation rule mining to detect factors which contribute to heart disease in males and females. Expert Syst Appl 2013;40(4). doi: https://doi.org/10.1016/j.eswa.2012.08.028.

[13] S. H. Ali, ''Miner for OACCR: Case of medical data analysis in knowledge discovery,'' IEEE, 2012 6th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), Sousse, 2012, pp. 962-975. Doi: 10.1109/SETIT.2012.6482043.

[14] Kavitha, D., Haritha, Padma, Y. (2021) ''Optimized Candidate Generation for Frequent Subgraph Mining in a Single Graph. In: Chaki, N., Pejas, J., Devarakonda, N., Rao Kovvur, R.M. (eds) Proceedings of International Conference on Computational Intelligence and Data Engineering. Lecture Notes on Data Engineering and Communications Technologies, vol 56. Springer, Singapore. doi.org/10.1007/978-981-15-8767-2_23.

[15] Li H, Sheu PCY. A scalable association rule learning heuristic for large datasets. J Big Data 2021;8:86. doi: https://doi.org/10.1186/s40537-021-00473-3.

[16] Lopamudra Dey, Sanjay Chakraborty, Anirban Mukhopadhyay.. (2020) ''Machine learning techniques for sequence-based prediction of viral–host interactions between SARS-CoV-2 and human proteins,'', Biomedical Journal., pp 438-450. 32. 10.1016/ j.bj.2020.08.003.

[17] Wang M, Song L, Zhang Y, Hongli Gao Lu, Yan BY. Malsite-Deep: Prediction of protein malonylation sites through deep learning and multi-information fusion based on NearMiss-2 strategy. Knowl-Based Syst 2022;vol 240 ISSN:0950–7051. doi: https://doi.org/10.1016/j.knosys.2022.108191.

[18] Nguyen LB, Nguyen LT, Zelinka I, Snasel V, Nguyen HS, Vo B. ''A method for closed frequent subgraph mining in a single large graph. IEEE Access 2021;9:165719–33.

[19] Patel, H., Bhatt, D., Shakhreliya, S., Lam, N. L., Maurya, R., & Singh, V (2021) ''An Introduction and Applications of Bioinformatics,'', In *Advances in Bioinformatics* , (pp. 1-14). Springer, Singapore.

[20] Al_Janabi S, Al_Shourbaji I, Salman MA. Assessing the suitability of soft computing approaches for forest fires prediction. Applied Computing and Informatics 2018;14(2):214–24.

[21] Rehman SU, Asghar S. ''Online social network trend discovery using frequent subgraph mining. Soc Netw Anal Min 2020;10:67. doi: https://doi.org/10.1007/s13278-020-00682-3.

[22] Al-Janabi S, Alkaim A. A novel optimization algorithm (Lion-AYAD) to find optimal DNA protein synthesis. Egyptian Informatics Journal 2022;23 (2):271–90. doi: https://doi.org/10.1016/j.eij.2022.01.004.

[23] Salloum, S. A., AlHamad, A. Q., Al-Emran, M., & Shaalan, K. (2018) ''A survey of Arabic text mining. In: *Intelligent Natural Language Processing: Trends and Applications* . (pp. 417-431). Springer, Cham.

[24] Al-Janabi S. Overcoming the Main Challenges of Knowledge Discovery through Tendency to the Intelligent Data Analysis. International Conference on Data Analytics for Business and Industry (ICDABI) 2021;2021:286–94. doi: https://doi.org/10.1109/ICDABI53623.2021.9655916.

[25] Singh, V., & Kumar, A. V (2021) ''An Introduction and Applications of Bioinformatics,'' *Advances in Bioinformatics*, Springer. DOI: 10.1007/978-981-33-6191-1_11.

[26] Vijayakumar V, Nedunchezhian RA. A study on video data mining. Int J Multimed Info Retr 2012;1:153–72. doi: https://doi.org/10.1007/s13735-012-0016-2.

[27] Kadhuim ZA, Al-Janabi S. Codon-mRNA prediction using deep optimal neurocomputing technique (DLSTM-DSN-WOA) and multivariate analysis. Results in Engineering 2023;17:100847.

[28] Zhang, C., Han, J. (2021) ''Data Mining and Knowledge Discovery. In: Shi, W., Goodchild, M.F., Batty, M., Kwan, MP., Zhang, A. (eds) Urban Informatics,'', The Urban Book Series, Springer, Singapore .Volume 142. https://doi.org/10.1007/978-981-15-8983-6_42.

[29] Zhao, G., Li, M., Jiang, Y. (2022) ''Prediction of Drug-Gene Interaction by Using Biomedical Subgraph Patterns. In: Chu, SC., Lin, J.CW., Li, J., Pan, JS. (eds) Genetic and Evolutionary Computing. ICGEC 2021. Lecture Notes in Electrical Engineering, vol 833. Springer, Singapore. https://doi.org/10.1007/978-981-16-8430-2_15.