An Algorithmic Solution for the Hair Ball Problem in Data Visualization

Khalid H. Alnafisah
Department of Computer and Information Science
Gannon University, 109 University Square, Erie, PA 16541
alnafisa001@gannon.edu
alnafisah.k.h@ieee.org

Abstract— The investigation and analysis of large and complex graphs is an important aspect of data visualization research, yet there is a need for entirely new, scalable approaches and methodologies for graph visualization. This can ultimately provide more insight into the structure and function of this complex graph.

To explain more, we to need find a methodology to develop a solution to present a "tidy" graph with the minimal crossover between edges in the "Hair Balls." In spite of the expanding significance of investigating and extensively analyzing and understanding very large graphs of data, the traditional way of visualizing graphs has difficulties scaling up, and typically ends up depicting these large graphs as "Hair Balls". This traditional approach does indeed have a deeply intuitive foundation: nodes are depicted with a shape such as a circle, triangle or square, which are then connected by lines or curves that represent the edges. In any case, although there are many different ways to apply this basic underlying idea, it needs to be revisited in light of current and emerging needs for understanding increasingly complex crossover between edges in the graphs. The complex "Hair Ball," which appears as an indecipherable graph, came from the crossover between edges.

From our preliminary research, we found the major disadvantage in the Hair Balls graph was that it confused observers. Users may think there are some extra nodes; but in reality, there are not. Because there are many crossovers between edges in the Hair Balls, the impression also may affect observers' understanding of the whole structure of the graph. Major problem-no effective reception of information from a "Hair Balls" graph-meaningless to observers.

Fig. 1. Social Media Graph.

Keyword— Bipartite $K_{2,3}$, Crossover, Algorithmic, Complex Gephi.

1. Introduction

1.1 Overview— In this research, we will study the questions, "What is the Hair Ball Problem?" and "What is the reason for the Hair Ball Problem?" Discovering answers to these questions will lead observers to understand the "Hair ball "problem in data visualization. In large graphs, some crossovers between edges are always unavoidable, for example, Fig.1. We will find that less crossovers in large graphs allows us to develop a solution (Complete Bipartite Graph $K_{2,3}$) to present a "tidy" graph (semi-tree structure). This problem is inspired by those recent "big data" research. The nature of this research will be about developing a new algorithm to provide a better layout for big graphs [1][27].

1.1 Curriculum Scope— In the curriculum Scope, the learning experience from my graduate-level GCIS 523 Statistical Computing was helpful in the progression of the project phases. It were worthy classes (GCIS 523 Statistical Computing, GCIS 644 Knowledge-Based Systems and GCIS 605 Scholarship Seminar) that I had during my school time (fall 2014, spring 2015 and fall 2015). For example, Statistical Computing Class. Statistical Computing Class was helpful for learning R Programing [22]. R. Programing is necessary in the construction of my research. The classes here necessary were for my research in data visualization and understanding how to answer my research questions. In this classes we deal with small and big dataset by manual calculation like my dataset (Media organizations dataset, AnAge dataset and Time Use). We have to use some of special programing to deal with these datasets like R Programing and Gephi. I found R programing is a good programing to my research special in the parts of Step 1(find and locate all the $K_{2,3}$ graph in a graph), Step 2(compressing the $K_{2,3}$ and create a new graph) and Step 3 (prune the new graph and try to separate those $K_{2,3}$ to we can see if the graph is tidy or still has a complex crossover between the edges). Gephi is to visualization the dataset which is for the step 4 (represent these $K_{2,3}$) [20].

The Knowled $K_{2,3}$ ge-Based Systems course was really helpful course when dealing with the data visualization phase on my research. With the experience from the course, I was able to represent these $K_{2,3}$ to I can see if the graph became a tidy graph or still not a tidy graph and plan the steps accordingly [21].

1.1 Identify the Problem Area— In large graphs, too many crossovers are unavoidable (Two edges cross each other on the graph presentation which may cause confusion for extra node) [4][6]. In more explain, one cannot avoid multiple crossovers in large graphic data. As shown in figure 2, crossovers are classic data visualization problem. The disadvantage is the original graph may confuse observers [13]. It is difficult to observe the structure of the graph. In wider explain, disadvantage include graph confusion due to an over whelming amount of crossovers, and an inability to decipher information [12] [32].

Fig. 2. Crossover in $K_{2,3}$

As you see in figure 2, observers may not be able to see node behind this crossover.

2. RESEARCH APPROACH

2.1 Hypothesis

- The "hair ball" problem in data visualization is entirely dependent on $K_{2,3}$.
- $K_{2,3}$**,** a discrete solution can be applied to a "Big Graph".

In this research, the algorithmic solutions will steps as follow [24] [29]:

- **Step 1:** we find and locate all the $K_{2,3}$ graph within a complex graph.
- **Step 2:** by compressing the $K_{2,3}$, we can create a new graph.
- **Step 3:** we prune the new graph separate the $K_{2,3}$ sections (expand the $K_{2,3}$ nodes to discern data), and so we can see if the graph is tidy or has a complex crossover between the edges.
- **Step 4:** represent these $K_{2,3}$.

Note, the description above is only a rough idea. More complicated structure, such as loop and embedded functions may be used in the real solutions.

3. TECHNICAL APPROACH

3.1 R Script [15][22] **or Gephi Visulization Tool** [14][30] **—**In this part, I used R program and Excel to plot the graph. I used R programing because R is an integrated suite of software facilities for data manipulation, calculation, and graphical display. To use R script you will download this program using the following outlined steps [23]:

1- Go to the http://www.r-project.org/
2- In the left chose the CRAN

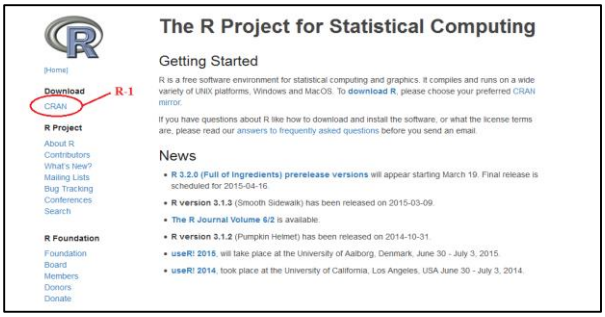

Fig. 3. Download R.

3.1.1 R packages In— this research, we will uses several key packages that you will need to install in order to follow along. Several other libraries will be mentioned along the way, but those are not critical and can be skipped. The main libraries we are going to use are igraph packages, bipartite packages and rgexf packages [15].

- For a Network Analysis and Visualization.
  # https://cran.r-project.org/web/packages/igraph/index.html
  install.packages("igraph", dependencies=TRUE)
  library(igraph) [2]

- Calculates a variety of indices and values for a bipartite network [8].
  # http://www.inside-r.org/packages/cran/bipartite/docs/.networklevel
  install.packages("bipartite")
  library(bipartite)

4. IMPLEMENTATION DESIGN (ALGORITHM)

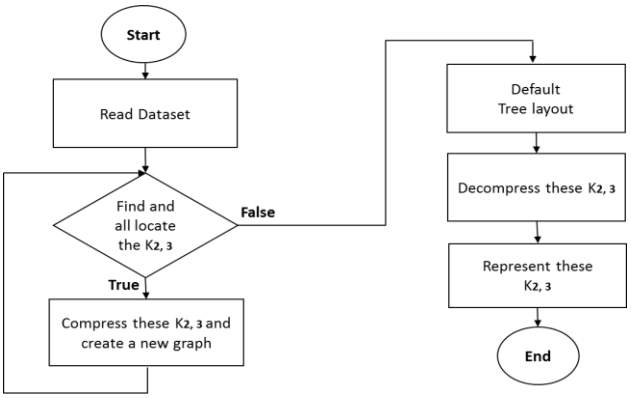

Fig. 4. Implementation design (Algorithm).

5. METHODOLOGY OF RESSEARCH

5.1 Objectives of the Research

- Study the "Hair Ball" problem in data visualization.

- Develop a solution to present a tidy graph with minimal crossover between edges in the Hair Balls.
- Test, evaluate and verify my solution on multiple (at least three) big and complex graph datasets.

5.2 Methods Workflow

5.2.1 Dataset

5.2.1.1 **Small Datasets—** There are four small dataset that we made-up it. These four small dataset with a $K_{2,3}$ or without any K **$K_{2,3}$**.

5.2.1.2 Big Dataset— There are three big dataset we will use it in this research. These three big dataset from different resources / categories. At least 100 nodes for each graph.

5.2.1.2.1 Big Datase 1-AnAge Dataset —— Animal Ageing and Longevity. This file contains only the raw data. My goal is to find any $K_{2,3}$ bipartite in the life history and lifespan.

5.2.1.2.2 Big Dataset 2-Media Organizations Dataset It is a network of links between media venues and consumers. We will rarely use certain visual properties such as the shape of the node symbols and $K_{2,3}$ Bipartite: those are impossible to distinguish in larger graph maps. In fact, when drawing very big networks we may even want to hide the network edges, and focus on $K_{2,3}$ bipartite of nodes. At this point, the size of the networks you can visualize in R is limited mainly by the RAM of your machine.

5.2.1.2.3 Big Dataset 3-Time Use Dataset— How people spend their time depending on country and sex, with activities such as paid work, household and family care, etc.

5.2.2 Workflow of Small Datasets 2 (One $K_{2,3}$ Bipartite)— An Excel file is used to create the dataset which have one $K_{2,3}$. Two columns are evident with seventeen rows in the Excel sheet. The name of the first column is "x" and the second column is called "y." I give the first three rows in the columns of "x" same number which is "1." Also, I give the second three rows in the columns of x same numbers but different than number "1" which is "2." In the opposite direction of the column "y," I give the first three rows in the columns three different numbers which is "3, 4 and 5." Also, I will give the second three rows in the column of "y" the same group in the first three rows in column "y" which is "3, 4 and 5." The rest of the rows in both columns we will give those randomly numbers but completely different than which we gave to the first six rows in both column as showing in the table 1.

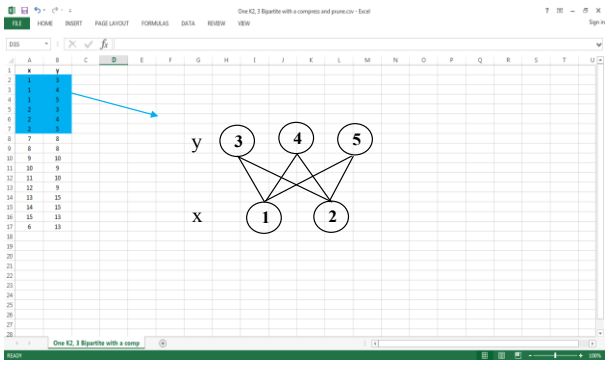

Table 1: One $K_{2,3}$ Bipartite.

The small dataset, there is one $K_{2,3}$ bipartite. By using the linkcomm package (with the library in R programing) you can check if the dataset has any $K_{2,3}$ Bipartite or not. But, before I use this package or any package I have to use the "igraph package with it library," for a Network Analysis and Visualization.

The linkcomm package, is a "provide tool" for the generation, visualization, and analysis of link communities in networks of arbitrary size and type. As shown in figure 5, there is one $K_{2,3}$ in the dataset, because when we use the linkcomm command we will have 5 nodes in the cluster which is these 5 nodes in cluster are one $K_{2,3}$ bipartite as shown in figure 5.

```
# linkcomm package
lc <- getLinkCommunities(SmallDatasets2,
hcmethod = "single")
```

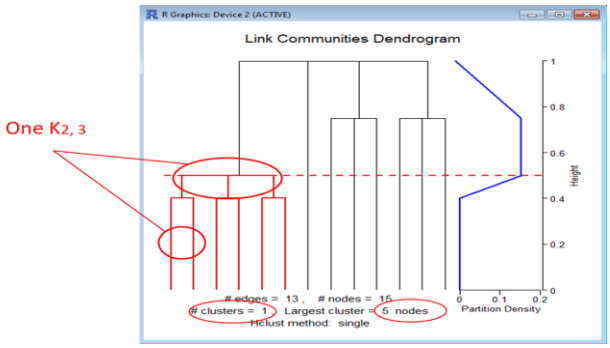

Fig. 5. One $K_{2,3}$ Bipartite by Linkcomm Package before Compress.

If the Linkcomm Package indicates a $K_{2,3}$ cluster, the next step would be to use the Bipartite Layout to calculate a variety of indices and values for a bipartite network [26]. As shown in figure 23, there is one $K_{2,3}$ because we have one cluster with 5 nodes which confirms this cluster is One $K_{2,3}$ Bipartite [18].

```
#Bipartite Package [7]&[16]
lc <- getLinkCommunities(CompressBipartite,
hcmethod = "single")
plot(igraph, mark.groups=c(1),
mark.col="#FFFF00",vertex.size=15,
layout=layout.bipartite,
vertex.label=V(igraph)$media, vertex.label.cex=.7,
main="One Bipartite K2, 3")
```

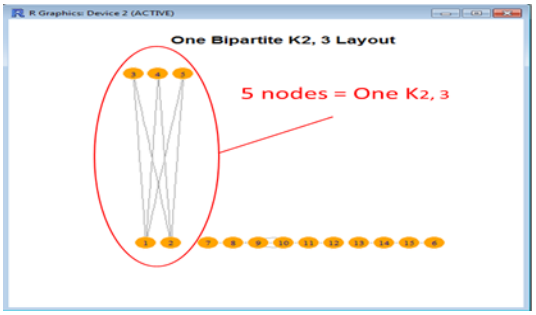

Fig. 6. One $K_{2,3}$ Bipartite by Bipartite Package before Compress.

After confirming the $K_{2,3}$ bipartite is ineligible for this dataset, we will use a special layout [9] to make this graph a "tidy" graph. The first step is to see how this dataset looks (visualization) by using a random layout as shown in figure 7 [26].

```
# Random Layout
plot(igraph, layout=layout.random,
main="Random  Layout")
```

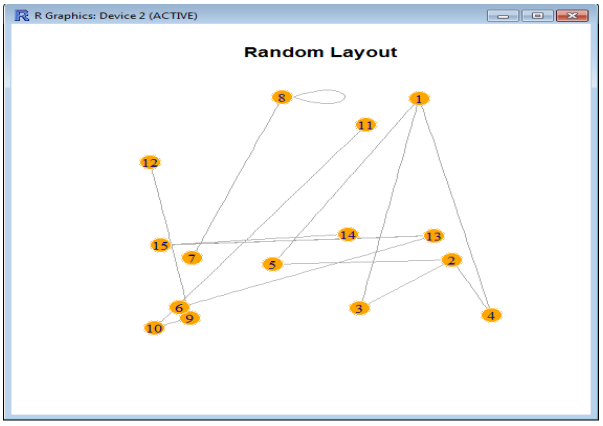

Fig. 7. Random Layout.

Figure 7 is not a tidy graph, because it has one $K_{2,3}$. After we find and locate this $K_{2,3}$, compressing (cluster) the next step is to will makes this graph a tidy graph by compressing the $K_{2,3}$ and creating a new graph. The following procedure is to assign all the nodes in the $K_{2,3}$ to one letter or number to be come all these nodes (5 node in one cluster) in the same number or letter. The next step will assign all the nodes in the $K_{2,3}$ to B Followed by a number and different color around the node than others node, for example "B1" with yellow color around the node.

After that, we will plot this graph to find this $K_{2,3}$ become one node not 5 node as we saw in the Tree layout [19] graph before we use this scenario, see figure 8.

```
Compress <- data.frame(SmallDatasets2)
Compress
attach(Compress)
Compress$x[1:6]= c("B1")
attach(Compress)
x
Compress$y[1:6]=c("B1")
attach(Compress)
y
attach(Compress)
Compress
write.csv(Compress, file =
"MyData.csv",row.names=FALSE)
CompressBipartite<- read.csv("MyData.csv",
header=T, as.is=T)
CompressBipartite
Igraph2 <- graph.data.frame(CompressBipartite)
Igraph2
V(igraph2)$type <- V(igraph2)$name %in%
CompressBipartite[,1]
bipartite.projection(igraph2)
bipartite_mapping(igraph2)
```

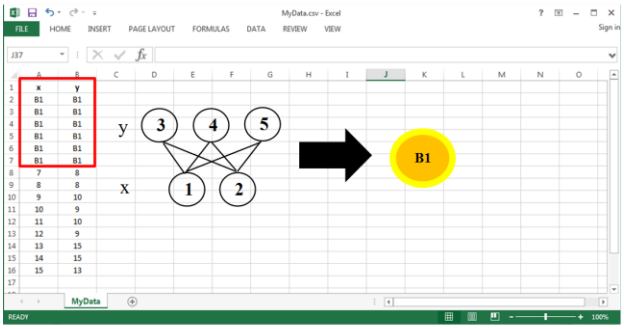

Table 2: Compress One $K_{2,3}$ Bipartite.

```
# fruchterman.reingold.layout(tree layout)
V(igraph2)$frame.color <- "white"
V(igraph2)$color = "orange"
V(igraph2)$size = 13
E(igraph2)$arrow.mode = 0
plot(igraph2, layout=layout.fruchterman.reingold,
main="Tree layout (One K2, 3 Bipartite)",
mark.groups=c(1), mark.col="#FFFF00")
legend(x=-1.5, y=-1.1, c("K2,3 Bipartite-1"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="#FFFF00")
```

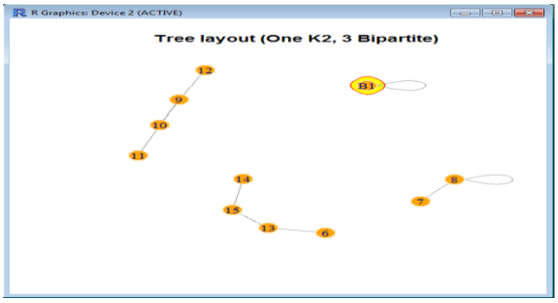

Fig. 8. Tree Layout after the Compress the $K_{2,3}$.

After compressing the $K_{2,3}$ and creating the new graph, the next step is to check if there is another $K_{2,3}$ in the graph by using the Linkcomm Package feature as shown in figure 9 [3].

```
lc <- getLinkCommunities(CompressBipartite,
hcmethod = "single")
```

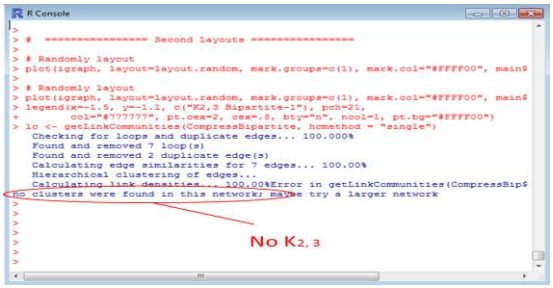

Fig. 9. No $K_{2,3}$ by Linkcomm Package after the compress the $K_{2,3}$.

As evident in figure 9, there are no more $K_{2,3}$ in the dataset, because all of them have been compressed. The figure indicates, “no clusters were found in this network”, which means there are no more $K_{2,3}$ in the dataset.
Verify the absence of $K_{2,3}$ bipartite, the next step is to use the bipartite package which calculates a variety of indices and values for a bipartite network. As shown in figure 10, there are no more $K_{2,3}$ because the nodes at the bottom cannot connect to nodes on the top of the figure.

```
#Bipartite Package
lc <- getLinkCommunities(CompressBipartite,
hcmethod = "single")
plot(igraph, mark.groups=c(1),
mark.col="#FFFF00",vertex.size=15,
layout=layout.bipartite,
vertex.label=V(igraph)$media, vertex.label.cex=.7,
main="No Bipartite K2, 3")
legend(x=-1.5, y=-1.1, c("K2,3 Bipartite-1"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="#FFFF00")
```

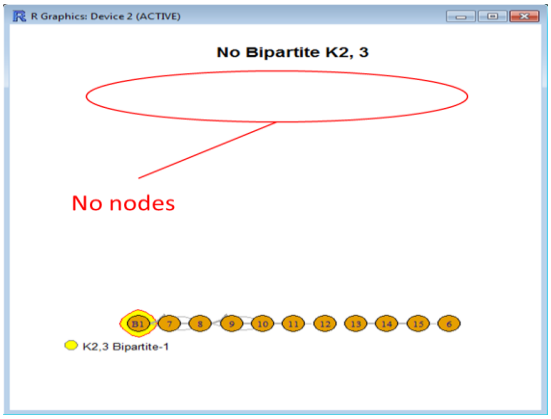

Fig. 10. Bipartite Package after the Compress the $K_{2,3}$.

In this dataset there is one $K_{2,3}$ in the graph. Last step in the algorithm steps of the hypothesis is to use "Represent $K_{2,3}$” through R or Gephi. To represent this $K_{2,3}$ by R, just selected and customized one of the available layout algorithms in it as shown in figure 11 [26].

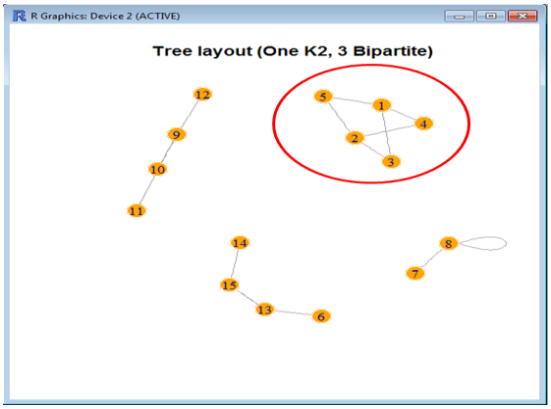

Fig. 11. Represent Graph by Tree Layout.

All the node in the interface of Gephi in the same color except one node. This one node (orange color) is one $K_{2,3}$. We compressed five nodes in one node. When we back to represent all the nodes, the only one it will be change is the node that has the $K_{2,3}$ (orange node) as shown in figure 12 and 13.

1. VALIDATION

1.1 Small Datasets

1.1.1 Small Dataset 1 (No $K_{2,3}$ Bipartite)— An Excel file is used to create the dataset which did not have a $K_{2,3}$. Two columns are evident with fifteen rows in the Excel sheet. The name of the first column is "x" and the second column is called "y." We give all the rows in both columns randomly numbers which do not have $K_{2,3}$ in this dataset as shown in the Table 3.

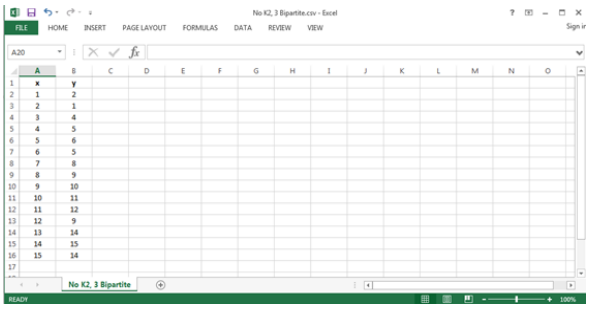

Table 3: No $K_{2,3}$ Bipartite.

The small datasets in excel do not contain $K_{2,3}$ Bipartite. By using the linkcomm package with the library in R programing, the dataset can be verified by check if the dataset has any $K_{2,3}$ Bipartite or not. The linkcomm package, is a tool that provides the generation, visualization, and analysis of link communities in networks of arbitrary size and type. Before using this package or any package, one must to use the “igraph package with it library,” for Network Analysis and Visualization. As shown in figure 12, there are no $K_{2,3}$ because there needs to be five nodes in the cluster, but in this figure we have just fore nodes in one cluster.

```
install.packages("igraph", dependencies=TRUE)
library(igraph)
install.packages("linkcomm")
library(linkcomm)
lc <- getLinkCommunities(SmallDatasets1,
hcmethod = "single")
```

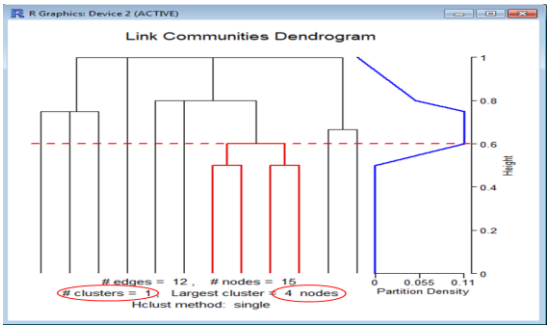

Fig. 12. No $K_{2,3}$ Bipartite by linkcomm package.

To make sure if the linkcomm package provides the correct result, a bipartite package is used to calculate a variety of indices and values for a bipartite network. As shown in figure 13, there are no $K_{2,3}$ because we have to have nodes facing (at the top) and connects to the nodes on the bottom of figure.

```
#Bipartite Package
lc <- getLinkCommunities(SmallDatasets1,
hcmethod = "single")
plot(igraph1, vertex.size=15,
layout=layout.bipartite,
vertex.label=V(igraph1)$media,
vertex.label.cex=.7, main="No Bipartite K2, 3 ")
```

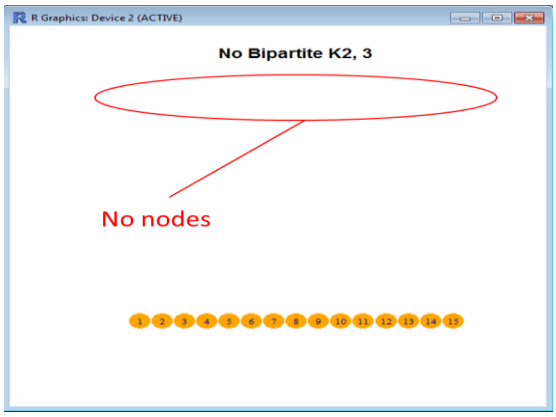

Fig. 13. No $K_{2,3}$ Bipartite by bipartite package.

After verifying that $K_{2,3}$ Bipartite is ineligible for this dataset, I will see how this dataset looks (visualization) by using a randomly layout as shown in figure 14.

```
# Random layout
plot(igraph1, layout=layout.random,
main="Random Layout")
```

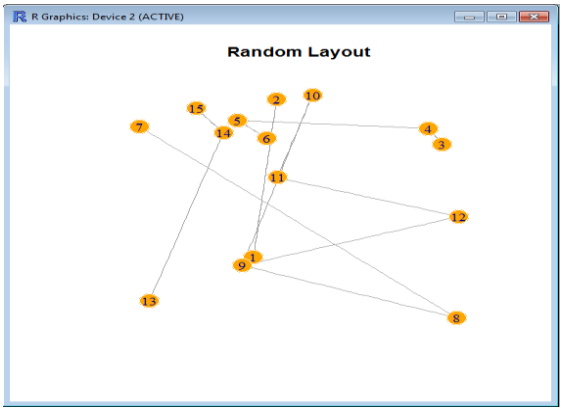

Fig. 14. No $K_{2,3}$ Bipartite by Random layout.

After that, One layout is used which is tree layout. Fruchterman Reingold Layout (Tree layout): The tree Layout is a force-directed layout algorithm. The idea of a force directed layout algorithm is to consider a force between any two nodes.

```
# fruchterman.reingold.layout (Tree Layout)
V(igraph1)$frame.color <- "white"
V(igraph1)$color = "orange"
V(igraph1)$size = 13
E(igraph1)$arrow.mode = 0
```

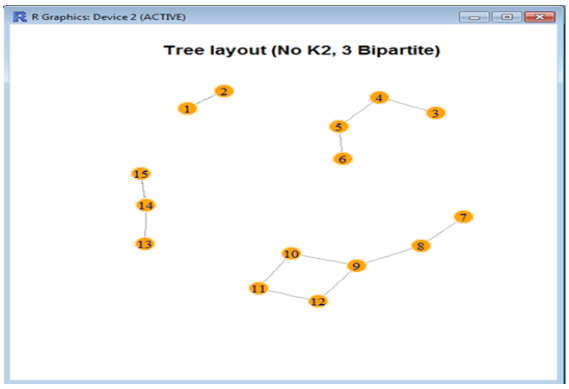

Fig. 15. Tree layout.

In this dataset there is no any $K_{2,3}$ in the graph. We did not use the last part of the algorimetic solutions of the hypothesis "Represent these $K_{2,3}$."

5.2.3 Small Dataset 2 (Two $K_{2,3}$ Bipartite)— An Excel file is used to create the dataset which two have **$K_{2,3}$**. Two columns are evident with eighteen rows in the Excel sheet. The name of the first column is "x" and the second column is call "y." The first three rows in the columns of "x" are labeled with the same number which is "1." Also, the second three rows in the columns of x are labeled with the same number but different than number "1" which is "2." In the opposite direction, which is column "y," provides the first three rows in the columns three different numbers which is "3, 4 and 5." Also, provides the second three rows in the column of "y" the same group in the first three rows in column "y" which is "3, 4 and 5." The second $K_{2,3}$ in this dataset we will give from row eleven to the row thirteen in the column of "x" same number which is" 11." Also, provides from row fourteen to the row sixteen in the column of x same numbers but different than number "11" which is "12." In the opposite direction which is column "y," provides from row eleven to the row thirteen three different numbers which is "13, 14 and 15." Also, provides from row fourteen to the row sixteen in the column of "y" the same group in the row eleven to the row thirteen which is "13, 14 and 15" as shown in the Table 4 [10].

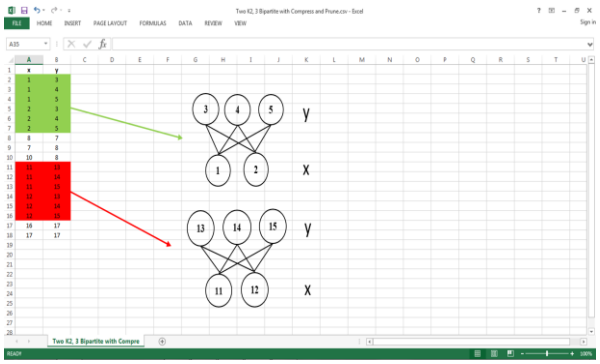

Table 4: Two $K_{2,3}$ Bipartite.

In small dataset, there are two $K_{2,3}$ bipartite. By using the linkcomm package with the library in R programing, the dataset can be verified by check if the dataset have $K_{2,3}$ Bipartite or not. Before using this package or any package, one must to use the "igraph package with it library," for a Network Analysis and Visualization. The linkcomm package is a "provide tool" for the generation, visualization, and analysis of link communities in networks of arbitrary size and type. As shown in figure 22, there are two $K_{2,3}$ in the dataset, because when used the linkcomm command, we will have two cluster with 5 nodes in each other. These 5 nodes in one cluster are one $K_{2,3}$ bipartite as shown in figure 16 [10].

```
# linkcomm package
lc <- getLinkCommunities(SmallDatasets3,
hcmethod = "single")
```

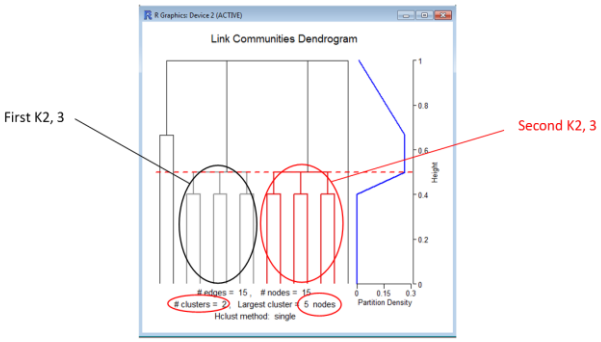

Fig. 16. Two $K_{2,3}$ Bipartite by Linkcomm Package before Compress.

If the Linkcomm Package indicates a $K_{2,3}$ cluster, the next step would be to use the Bipartite Package to calculate a variety of indices and values for a bipartite network. As shown in figure 18, there is two $K_{2,3}$ because that have two clusters with 5 nodes for each which confirms these clusters are "Two $K_{2,3}$ Bipartite."

```
#bipartite Package
lc <- getLinkCommunities(SmallDatasets3,
hcmethod = "single")
plot(igraph1, vertex.size=15,
layout=layout.bipartite,
vertex.label=V(igraph1)$media,
vertex.label.cex=.7, main="Two Bipartite K2, 3")
```

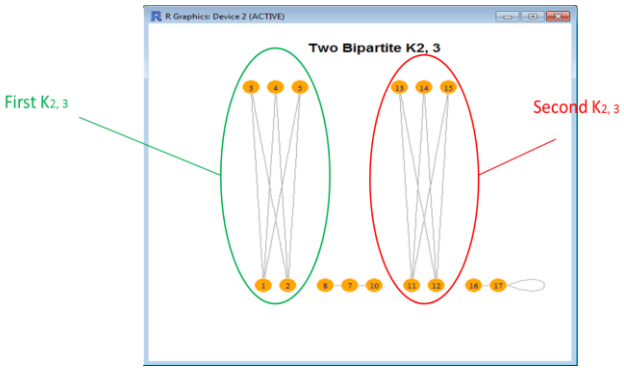

Fig. 17. Two $K_{2,3}$ Bipartite by Bipartite Package before Compress.

After verifying that $K_{2,3}$ bipartite is ineligible for this dataset, used a special layout to make this graph a "tidy" graph. The first step is to see how this dataset looks (visualization) by using a Random Layout as shown in figure 18.

```
# Random layout
plot(igraph1, layout=layout.random,
main="Random Layout")
```

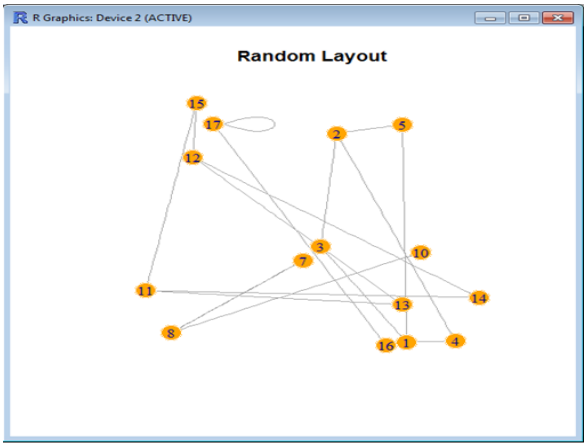

Fig. 18. Random Layout.

Figure 19 is not a tidy graph, because it has two $K_{2,3}$. After find and locate this $K_{2,3}$, compress (cluster) the next step is to makes this graph a tidy graph by compressing those $K_{2,3}$ and creating a new graph. The following procedure is to assign all the nodes in the $K_{2,3}$ to one letter or number to be come all these nodes (5 node in one cluster) in the same number or letter. The next step is to assign all the nodes in the first $K_{2,3}$ to B followed by a number and different color around the node than others node. For example, "B1" with green color [28] around the node for the first $K_{2,3}$ and red color around the node for the second $K_{2,3}$ with the "B2." After that, ploted this graph to find these $K_{2,3}$ become two nodes not ten (five for every $K_{2,3}$ cluster) nodes as we see in the randomly layout graph before we use this scenario, see figure 19.

```
Compress1 <- data.frame(SmallDatasets3)
Compress1
attach(Compress1)
Compress1$x[1:6]= c("B1")
Compress1$x[10:15]= c("B2")
attach(Compress1)
x
Compress1$y[1:6]=c("B1")
Compress1$y[10:15]=c("B2")
attach(Compress1)
y
attach(Compress1)
Compress1
write.csv(Compress1, file =
"MyData.csv",row.names=FALSE)
CompressBipartite1<- read.csv("MyData.csv",
header=T, as.is=T)
CompressBipartite1
```

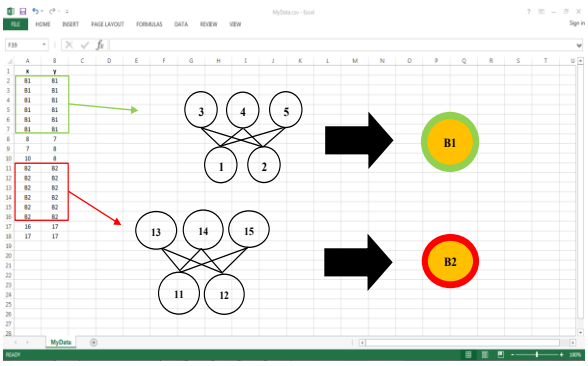

Table 5: Two $K_{2,3}$ Bipartite after the Compress.

```
# fruchterman.reingold.layout(Tree layout)
V(igraph2)$frame.color <- "white"
V(igraph2)$color = "orange"
V(igraph2)$size = 13
E(igraph2)$arrow.mode = 0
plot(igraph2,
layout=layout.fruchterman.reingold,mark.groups=li
st(c("B1"),c("B2")),mark.col=c("green","red"),
main="Tree layout (Two K2, 3 Bipartite)")
legend(x=-1.5, y=-1.1, c("K2,3 Bipartite-1"),
pch=21,col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="green")
legend(x=-1.5, y=-1.2, c("K2,3 Bipartite-2"),
pch=21,col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="red")
```

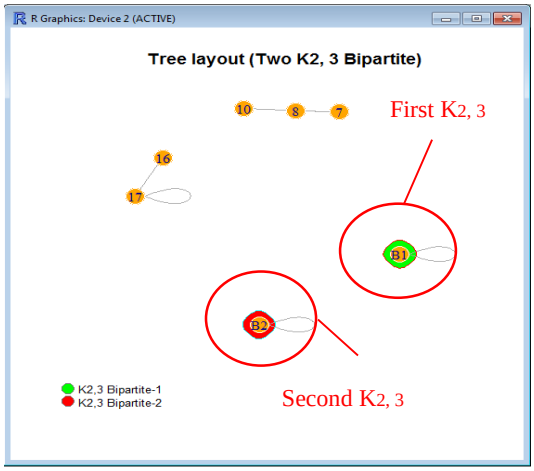

Fig. 19. Tree Layout after the Compress the K 2, 3.

After compressing the $K_{2,3}$ and creating the new graph, the next step is to check if there is another $K_{2,3}$ in the graph by using the Linkcomm Package feature as shown in figure 20.

```
lc <- getLinkCommunities(CompressBipartite,
hcmethod = "single")
```

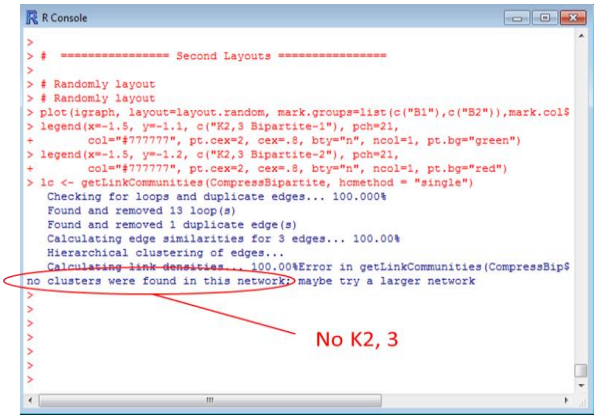

Fig. 20. No $K_{2,3}$ by Linkcomm Package after the compress the $K_{2,3}$.

As evident in figure 20, there are no more $K_{2,3}$ in the dataset, because all of them have been compressed in the graph. The figure indicates, "no clusters were found in this network", which means there are no more $K_{2,3}$ in the dataset.

To verify the absence of $K_{2,3}$ bipartite, the next step is to use the bipartite package which calculates a variety of indices and values for a bipartite network. As shown in figure 21, there are no more $K_{2,3}$ because the nodes at the bottom cannot connect to nodes on the top of the figure.

```
#Bipartite layout
lc <- getLinkCommunities(CompressBipartite1,
hcmethod = "single")
plot(igraph2,vertex.size=15,
layout=layout.bipartite,
mark.groups=list(c("B1"),c("B2")),mark.col=c("gre
en","red"), vertex.label=V(igraph2)$media,
vertex.label.cex=.7,  main="No Bipartite K2, 3")
legend(x=-1.5, y=-1.1, c("K2,3 Bipartite-1"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="green")
legend(x=-1.5, y=-1.2, c("K2,3 Bipartite-2"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="red")
```

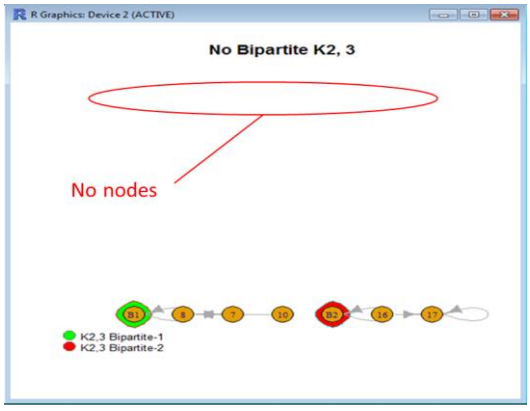

Fig. 21. No $K_{2,3}$ Bipartite by Bipartite Package.

In this dataset there are two $K_{2,3}$ in the graph. We used the last part of the algorimetic solutions of the

hypothesis "Represent $K_{2,3}$" through R. To represent these $K_{2,3}$ by R, we selected the Tree layout as shown in figure 22.

```
# fruchterman.reingold.layout(Tree layout)
V(igraph1)$frame.color <- "white"
V(igraph1)$color = "orange"
V(igraph1)$size = 13
E(igraph1)$arrow.mode = 0
plot(igraph1, layout=layout.fruchterman.reingold,
main="Tree layout (Two K2, 3 Bipartite)")
```

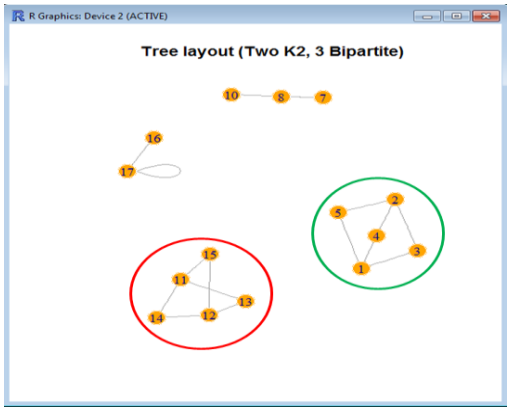

Fig. 22. Represent the Graph by Tree layout.

We see all the node in the interface of Gephi in the same color except two nodes. The green and red colored circle are examples of $K_{2,3}$ bipartite. Originally, there were ten nodes which were compressed into two $K_{2,3}$ bipartte. If the $K_{2,3}$ bipartite were to expand, only the $K_{2,3}$ (green and red colors) as shown in figure 24 and 25, would change shape.

1.1.1 Small Dataset 4 (Five $K_{2,3}$ Bipartite)— An Excel file is used to create the dataset which have five $K_{2,3}$ bipartite. There are two columns with 37 rows in the Excel sheet. The name of the first column is "x" and the second column is called "y." The first three rows in the columns of "x" same number which is "1." Also, the second three rows in the columns of "x" same numbers are labeled "2." In the column "y," the first three rows in the columns three are labeled as "3, 4 and 5." Also, the second three rows in the column of "y" the same group in the first three rows in column "y" which is "3, 4 and 5."In the second $K_{2,3}$ in this dataset labeled give the third three rows in the columns of "x" same number which is "6." Also, give the fourth three rows in the columns of x same numbers but different than number "6" which is "7." In the opposite direction which is column "y," We will give the third three rows in the columns three different numbers which is "8, 9 and 10." Also, we will give the fourth three rows in the column of "y" the same group in the third three rows in column "y" which is "8, 9 and 10." In the third $K_{2,3}$ in this dataset we will give the five three rows in the columns of "x" same number which is "11." Also, give the sixth three rows in the columns of x same numbers but different than number "11" which is "12." In the second column which is column "y," we will give the five three rows in the columns three different numbers which is "13, 14 and 15." Also, give the sixth three rows in the column of "y" the same group in the five three rows in column "y" which is "13, 14 and 15."

In the fourth $K_{2,3}$ in this dataset we will give the seventh three rows in the columns of "x" same number which is "16." Also, will give the eighth three rows in the columns of x same numbers but different than number "16" which is "17." In the opposite direction which is column "y," we will give the seventh three rows in the columns three different numbers which is "18, 19 and 20." Also, we will give the eighth three rows in the column of "y" the same group in the seventh three rows in column "y" which is "18, 19 and 20."

In the fourth $K_{2,3}$ in this dataset we will give the ninth three rows in the columns of "x" same number which is "21." Also, we will give the tenth three rows in the columns of x same numbers but different than number "21" which is "22." In the second column which is column "y," we will give the ninth three rows in the columns three different numbers which is "23, 24 and 25." Also, we will give the tenth three rows in the column of "y" the same group in the ninth three rows in column "y" which is "23, 24 and 25."

Between these $K_{2,3}$ there is $K_{2,3}$. We chose one node from every $K_{2,3}$ that we explained in the previous to make a $K_{2,3}$ between these five $K_{2,3}$. We will give the eleventh three rows in the columns of "x" same number which is "17." Also, we will give the twelfth three rows in the columns of x same numbers, but different than number "17" which is "22." In the second column which is column "y," we will give the eleventh three rows in the columns three different numbers which is "5, 10 and 15." Also, we will give the tenth three rows in the column of "y" the same group in the ninth three rows in column "y" which is "5, 10 and 15." This is the Scenario of the algorithmic on this dataset as shown in the figure 23.

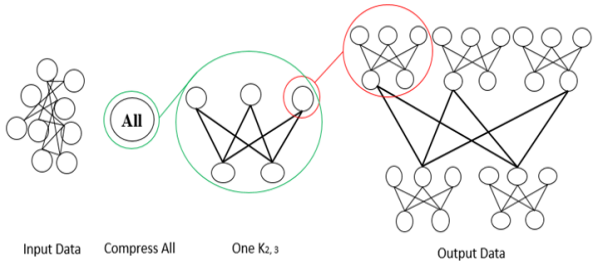

Fig. 23. Algorithm scenario on the dataset.

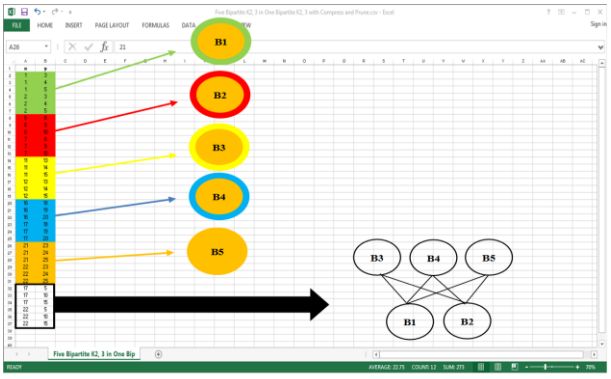

Table 6: Five Bipartite $K_{2,3}$ in One Bipartite $K_{2,3}$.

In the small dataset, there are six $K_{2,3}$ bipartite. By using the linkcomm package (with the library in R programing) the prevalence of $K_{2,3}$ bipartite can be verified. Before this package, or any package, are used the "igraph package with it library," Network Analysis and Visualization. The linkcomm package is a "provide tool" for the generation, visualization, and analysis of link communities in networks of arbitrary size and type. As shown in figure 27, there are six $K_{2,3}$ in the dataset (5 $K_{2,3}$ and one $K_{2,3}$ between those $K_{2,3}$), because when we use the linkcomm command, we will have six clusters each containing nodes. These 5 nodes in one cluster are one $K_{2,3}$ bipartite.

```
# linkcomm package
lc <- getLinkCommunities(SmallDatasets4,
hcmethod = "single")
```

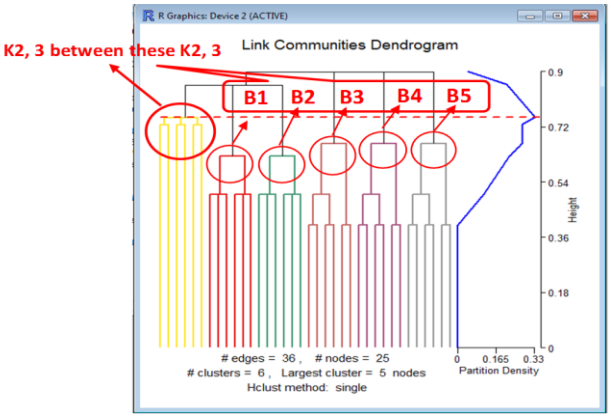

Fig. 24. Six $K_{2,3}$ Bipartite by Linkcomm Package before Compress.

If the Linkcomm Package indicates a $K_{2,3}$ cluster, the next step is to see how this dataset looks (visualization) by using a Random Layout as shown in figure 25.

```
# Random layout
plot(igraph1, layout=layout.random,
main="Random Layout")
```

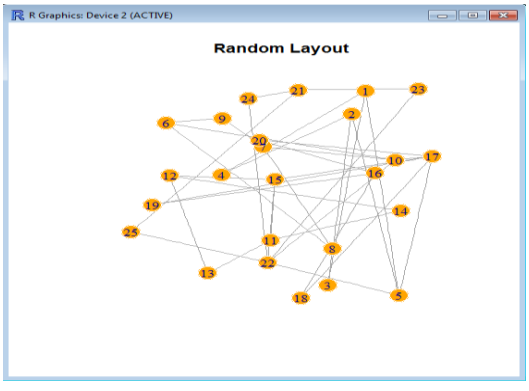

Fig. 25. Random Layout.

Figure 24 is not a tidy graph, because it has six $K_{2,3}$. After we find and locate these $K_{2,3}$, compress (cluster) the next step is to makes this graph a tidy graph by compressing these $K_{2,3}$ and creating a new graph. The following procedure is to assign all the nodes in the $K_{2,3}$ to one letter or number to be come all these nodes (5 node in one cluster) in the same number or letter. The next step is to assign all the compressing nodes in the first $K_{2,3}$ to B followed by a number and different color around the compressing node than others node, for example "B1" with green color around the compressing node for the first $K_{2,3}$. The red is assigned to the compressing node for the second $K_{2,3}$ with the "B2." Yellow color is assigned to the compressing node for the third $K_{2,3}$ with the "B3." Blue color is assigned around the compressing node for the fourth $K_{2,3}$ with the "B4." Orange color is assigned to the compressing node for the $K_{2,3}$ with the "B5" as shown in figure 26 [5].

```
Compress1 <- data.frame(SmallDatasets4)
Compress1
attach(Compress1)
Compress1$x[1:3]= c("B1")
Compress1$x[4:6]= c("B2")
Compress1$x[7:36]= c("B1")
attach(Compress1)
x
Compress1$y[1:1]= c("B3")
Compress1$y[2:2]= c("B4")
Compress1$y[3:3]= c("B5")
Compress1$y[4:4]= c("B3")
Compress1$y[5:5]= c("B4")
Compress1$y[6:6]= c("B5")
Compress1$y[7:36]= c("B1")
attach(Compress1)
y
attach(Compress1)
Compress1
write.csv(Compress1, file =
"MyData.csv",row.names=FALSE)
CompressBipartite1<- read.csv("MyData.csv",
header=T, as.is=T)
CompressBipartite1
igraph2 <- graph.data.frame(CompressBipartite1)
igraph2
V(igraph2)$type <- V(igraph2)$name %in%
Compress1[,1]
bipartite.projection(igraph2)
bipartite_mapping(igraph2)
lc <- largest.cliques(igraph2)
lc
sgc <- spinglass.community(igraph2)
sgc
# Randomly layout
plot(igraph2, layout=layout.random,
mark.groups=list(c("B1"),c("B2"),c("B3"),c("B4"),
c("B5")),mark.col=c("green","red", "yellow",
"blue", "orange"), main="Randomly Layout")
legend(x=-1.5, y=-1.1, c("K2,3 Bipartite-1"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="green")
legend(x=-1.5, y=-1.2, c("K2,3 Bipartite-2"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="red")
legend(x=-1.5, y=-1.3, c("K2,3 Bipartite-3"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="yellow")
legend(x=-1.5, y=-1.4, c("K2,3 Bipartite-4"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="blue")
```

```
legend(x=-1.5, y=-1.5, c("K2,3 Bipartite-5"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="orange")
```

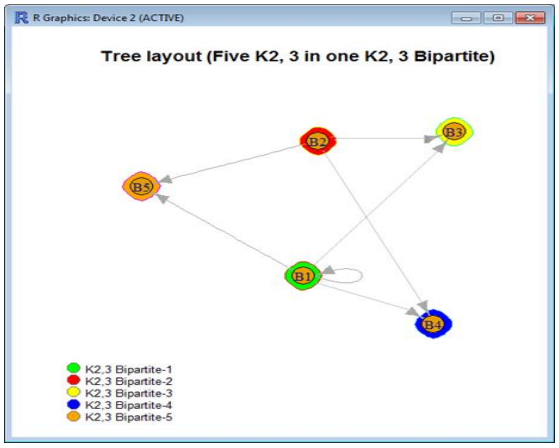

Fig. 26. Tree Layout- First Compress the $K_{2,3}$.

After compressing the $K_{2,3}$ and creating the new graph, the next step is to check if there is another $K_{2,3}$ in the graph by using the Linkcomm Package feature as shown in figure 27.

```
getLinkCommunities(CompressBipartite1,
hcmethod = "single")gle")
```

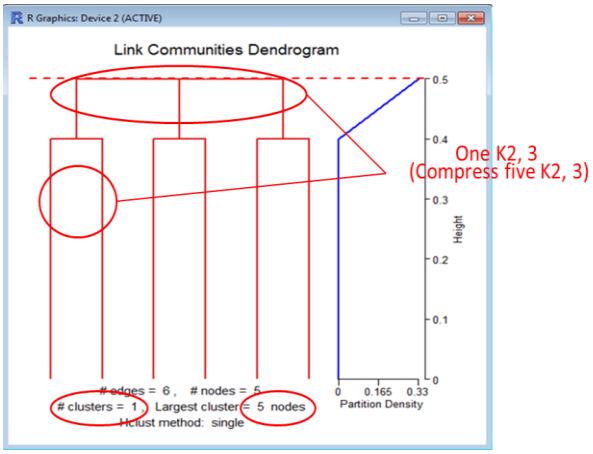

Fig. 27. Linkcomm Package- First Compress the $K_{2,3}$.

If the Linkcomm Package indicates a $K_{2,3}$ cluster, the next step would be to use the Bipartite Package to calculate a variety of indices and values for a bipartite network. As shown in figure 31, there is one $K_{2,3}$ because we have one clusters with 5 nodes for each which confirms these clusters are "One $K_{2,3}$ Bipartite."

```
# Bipartite K2, 3 Layout
lc <- getLinkCommunities(CompressBipartite1,
hcmethod = "single")
plot(igraph2, layout=layout.bipartite,
mark.groups=list(c("B1"),c("B2"),c("B3"),c("B4"),
c("B5")),mark.col=c("green","red", "yellow",
"blue", "orange"),main="Bipartite K2, 3 Layout")
```

```
legend(x=-1.5, y=-1.1, c("K2,3 Bipartite-1"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="green")
legend(x=-1.5, y=-1.2, c("K2,3 Bipartite-2"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="red")
legend(x=-1.5, y=-1.3, c("K2,3 Bipartite-3"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="yellow")
legend(x=-1.5, y=-1.4, c("K2,3 Bipartite-4"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="blue")
legend(x=-1.5, y=-1.5, c("K2,3 Bipartite-5"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="orange")
```

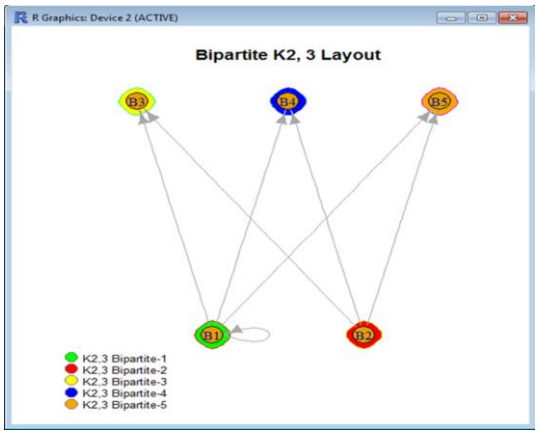

Fig. 28. Bipartite Package -First Compress the $K_{2,3}$.

Figure 28 is not a tidy graph, because it has one $K_{2,3}$. After we find and locate this $K_{2,3}$, compress (cluster) the next step is to makes this graph a tidy graph by compressing this $K_{2,3}$ and creating a new graph. The following procedure is to assign all the nodes in the $K_{2,3}$ to one letter, number or word to be come all these nodes (5 node in one cluster) in the same number, letter or word. The next step is to assign all the compressing nodes in the first $K_{2,3}$ to "All" with a color around the compressing node, for example, "All" with yellow color around the compressing node which meant this node call "All" is mean: B1, B2, B3, B4 and B5 as shown in figure 29.

```
compress2 <- data.frame(SmallDatasets4)
Compress2
attach(Compress2)
Compress2$x[1:36]= c("All")
attach(Compress2)
x
Compress2$y[1:36]=c("All")
attach(Compress2)
y
attach(Compress2)
Compress2
write.csv(Compress2, file =
"MyData.csv",row.names=FALSE)
CompressBipartite2<- read.csv("MyData.csv",
header=T, as.is=T)
CompressBipartite2
```

```
igraph3 <- graph.data.frame(CompressBipartite2)
igraph3
V(igraph3)$type <- V(igraph3)$name %in%
Compress2[,1]
bipartite.projection(igraph3)
bipartite_mapping(igraph3)
# Fruchterman.reingold.layout(tree layout)
V(igraph3)$frame.color <- "white"
V(igraph3)$color = "orange"
V(igraph3)$size = 13
E(igraph3)$arrow.mode = 0
plot(igraph3, layout=layout.fruchterman.reingold,
main="Tree layout (Five K2, 3 in one K2, 3
Bipartite)",mark.groups=c("All"),mark.col="#FFFF
00")
legend(x=-1.5, y=-1.1, c("K2,3 Bipartite-All"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="#FFFF00")
```

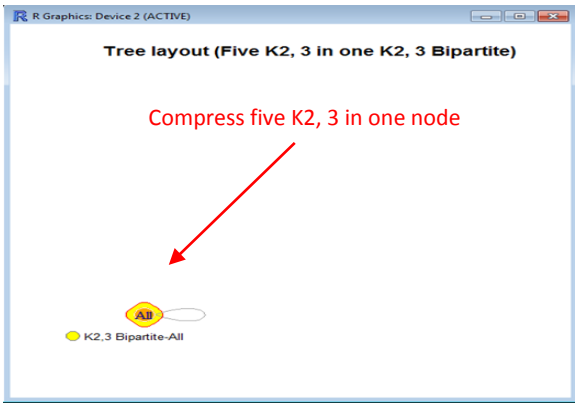

Fig. 29. Tree Layout- Second Compress the $K_{2,3}$.

After compressing the $K_{2,3}$ and creating the new graph, the next step is to check if there is another $K_{2,3}$ in the graph by using the Linkcomm Package feature as shown in figure 30.

```
lc <- getLinkCommunities(CompressBipartite,
hcmethod = "single")
```

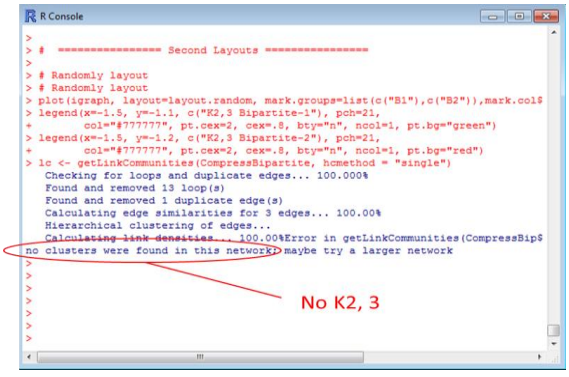

Fig. 30. No $K_{2,3}$ by Linkcomm Package- Second Compress the $K_{2,3}$.

Before the last step (Represent the $K_{2,3}$ by Gephi), we will change the position of the node as shown in figure 31.

```
V(igraph3)$size <- 20
# mar: A numerical vector of the form c(bottom,
left, top, right) which gives the number of lines of
margin to be specified on the four sides of the plot.
par(mar = c(14,13,0.0,0.0))
plot(igraph3, layout=layout.random,
mark.groups=c("All"),mark.col="#FFFF00")
```

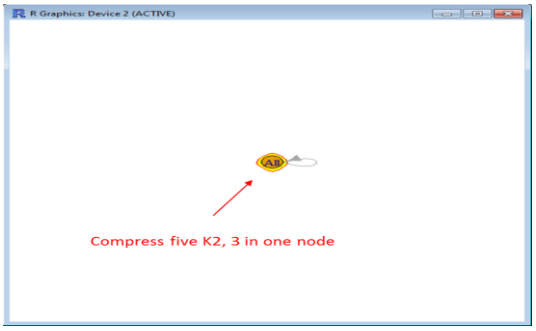

Fig. 31. Change the Position of the Node.

In this dataset there are five $K_{2,3}$ in one $K_{2,3}$ in the graph. The last part of the algorithm solutions of the hypothesis "Represent $K_{2,3}$" is performed through Gephi. To represent these $K_{2,3}$ by R, we selected and customized one of the available layout which is Tree Layout as shown in figure 32 and 33.

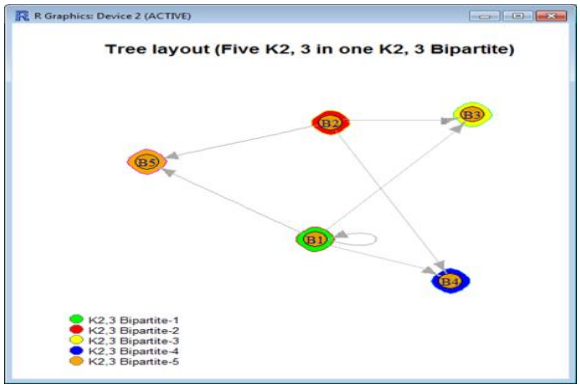

Fig. 32. First Represent Graph by Tree Layout.

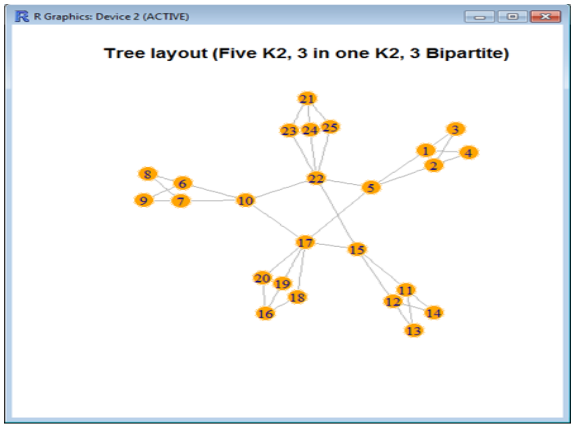

Fig. 33. Second Represent Graph by Tree Layout.

5.3 Big Datasets

5.3.1 Big Dataset 1-AnAge (No $K_{2,3}$ Bipartite)— The Big datasets of AnAge in excel do not contain $K_{2,3}$ Bipartite. By using the linkcomm package with the library in R programing, the dataset can be verified by check if the dataset has any $K_{2,3}$ Bipartite or not. The linkcomm package, is a tool that provides the generation, visualization, and analysis of link communities in networks of arbitrary size and type. As shown in figure 34 there are no $K_{2,3}$ because the message in R which is "no clusters were found in this network" verifies the absence of **$K_{2,3}$** Bipartite because there are no cluster node in this big dataset (AnAge).

```
linkcomm package
lc <- getLinkCommunities(AnAge, hcmethod =
"single")
```

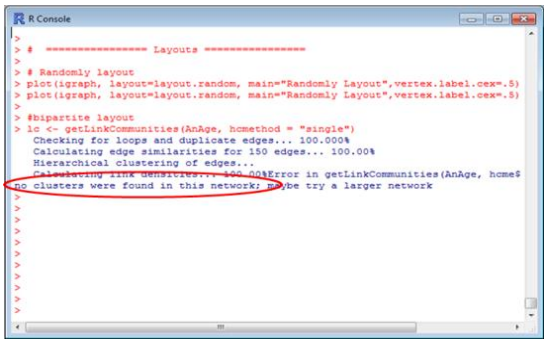

Fig. 34. No $K_{2,3}$ by Linkcomm Package.

After confirming the $K_{2,3}$ bipartite is ineligible for this dataset, we will use a special layout to make sure if this graph a "tidy" graph or not by using a Random Layout as shown in figure 35.

```
# Randomly layout
plot(igraph, layout=layout.random,
main="Randomly Layout",vertex.label.cex=.5)
```

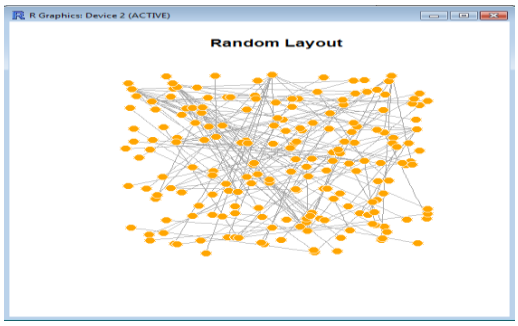

Fig. 35. Random Layout.

Because there is no $K_{2,3}$ in this dataset, we will use a special layout to make this graph a "tidy" graph by using a Tree Layout as shown in figure 36.

```
# fruchterman.reingold.layout(tree layout)
minC <- rep(-Inf, vcount(igraph))
maxC <- rep(Inf, vcount(igraph))
minC[1] <- maxC[1] <- 0
co <- layout_with_fr(igraph, minx=minC,
maxx=maxC,miny=minC, maxy=maxC)
co[1,]
plot(igraph, layout=co, vertex.size=60,
edge.arrow.size=0.2,vertex.label=c("", rep("",
vcount(igraph)-1)),
rescale=FALSE,xlim=range(co[,1]),
ylim=range(co[,2]), vertex.label.dist=0,
vertex.label.color="red",main="Tree layout (No
K2, 3 Bipartite)")
```

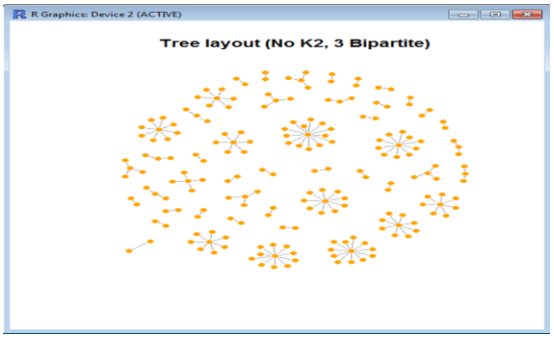

Fig. 36. Tree layout.

In this dataset, $K_{2,3}$ are not detected by the Linkcomm Package. We did not use the last part of the algorimetic solutions of the hypothesis "Represent these $K_{2,3}$."

5.3.2 Big Dataset 2-Media Organizations (14 $K_{2,3}$ Bipartite)— The Big datasets of Media Organizations in excel contains 14 **$K_{2,3}$** Bipartite. Some of the initial **$K_{2,3}$** are compressed into a secondary **$K_{2,3}$**. By using the linkcomm package with the library in R programing, the dataset can through checking the dataset for **$K_{2,3}$** bipartite presence. The linkcomm package, is a tool that provides the generation, visualization, and analysis of link communities in networks of arbitrary size and type. But, before I use this package or any package I have to use the "igraph package with it library," for a Network Analysis and Visualization must be used. In figure 37, there are 14 $K_{2,3}$ in this dataset, because using the linkcomm command, containing 16 clusters (14 $K_{2,3}$ and two $K_{2,3}$ between some of those clusters) with 5 nodes in each other. These 5 nodes in one cluster are one $K_{2,3}$ bipartite as shown in figure 37.

```
lc <- getLinkCommunities(Media, hcmethod =
"single")
```

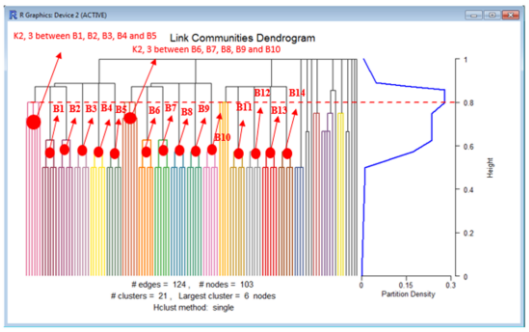

Fig. 37. 14 $K_{2,3}$ Bipartite by Linkcomm Package before Compress.

If the Linkcomm Package indicates a $K_{2,3}$ cluster, the next step would be to use the Bipartite Package to

calculate a variety of indices and values for a bipartite network. As shown in figure 18, there is two $K_{2,3}$ because that have two clusters with 5 nodes for each which confirms these clusters are "14 $K_{2,3}$ Bipartite."

```
#Bipartite Package
plot(igraph1, vertex.size=15,
layout=layout.bipartite,
vertex.label=V(igraph1)$media,
vertex.label.cex=.9, main="Bipartite K2, 3 Layout")
```

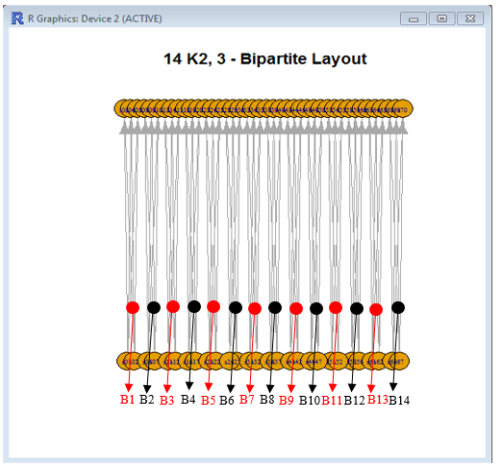

Fig. 38. Bipartite Package.

After confirming the $K_{2,3}$ bipartite is ineligible for this dataset, we will use a special layout to make sure if this graph a "tidy" graph or not by using a Random Layout as shown in figure 39.

```
# Random layout
V(igraph1)$frame.color <- "white"
V(igraph1)$color = "orange"
V(igraph1)$size = 11
E(igraph1)$arrow.mode = 0
plot(igraph1, layout=layout.random,
main="Random Layout",vertex.label.cex=.4)
```

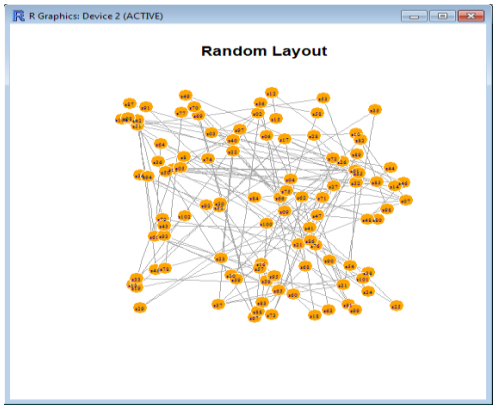

Fig. 39. Random Layout.

Figure 39 is not a tidy graph, because it has 14 $K_{2,3}$. After we find and locate these $K_{2,3}$, compress (cluster) the next step is to makes this graph a tidy graph by compressing these $K_{2,3}$ and creating a new graph. The following procedure is to assign all the nodes in the $K_{2,3}$ to one letter or number to be come all these nodes (5 node in one cluster) in the same number or letter. The next step is to assign all the compressing nodes in the first $K_{2,3}$ to the letter B followed by a number and different color around the compressing node than others node, for example "B1" with a green color around the compressing node for the first $K_{2,3}$. Red color is assigned around the compressing node for the second $K_{2,3}$ with the "B2." Yellow color is assigned around the compressing node for the third $K_{2,3}$ with the "B3." Blue color is assigned around the compressing node for the fourth $K_{2,3}$ with the "B4." Orange color is assigned around the compressing node for the fifth $K_{2,3}$ with the "B5." Cyan color is assigned around the compressing node for the fifth $K_{2,3}$ with the "B6." Firebrick color is assigned around the compressing node for the fifth $K_{2,3}$ with the "B7." Black color is assigned around the compressing node for the fifth $K_{2,3}$ with the "B8." Brown color is assigned around the compressing node for the fifth $K_{2,3}$ with the "B9." Magenta color is assigned around the compressing node for the fifth $K_{2,3}$ with the "B10." Tomato color is assigned around the compressing node for the fifth $K_{2,3}$ with the "B11." Pink color is assigned around the compressing node for the fifth $K_{2,3}$ with the "B12." Salmon color is assigned around the compressing node for the fifth $K_{2,3}$ with the "B13." Purple color is assigned around the compressing node for the fifth $K_{2,3}$ with the "B14" as shown in figure 40.

```
Compress1 <- data.frame(Media)
Compress1
attach(Compress1)
Compress1$from[1:3]= c("B1") # K2,3 Bipartite 1
Compress1$from[4:6]= c("B1")
Compress1$from[7:9]= c("B2") # K2,3 Bipartite 2
Compress1$from[10:12]= c("B2")
Compress1$from[13:15]= c("B1") # K2,3 Bipartite
Between All_1
Compress1$from[16:18]= c("B2")
Compress1$from[19:21]= c("B3") # K2,3 Bipartite 3
Compress1$from[22:24]= c("B3")
Compress1$from[25:27]= c("B4") # K2,3 Bipartite 4
Compress1$from[28:30]= c("B4")
Compress1$from[31:33]= c("B6") # K2,3 Bipartite
Between All_2
Compress1$from[34:36]= c("B7")
Compress1$from[37:39]= c("B5") # K2,3 Bipartite 5
Compress1$from[40:42]= c("B5")
Compress1$from[43:45]= c("B6") # K2,3 Bipartite 6
Compress1$from[46:48]= c("B6")
Compress1$from[49:51]= c("B7") # K2,3 Bipartite 7
Compress1$from[52:54]= c("B7")
Compress1$from[55:57]= c("B8") # K2,3 Bipartite 8
Compress1$from[58:60]= c("B8")
Compress1$from[65:67]= c("B9") # K2,3 Bipartite 9
Compress1$from[68:70]= c("B9")
Compress1$from[71:73]= c("B10") # K2,3 Bipartite 10
Compress1$from[74:76]= c("B10")
Compress1$from[77:79]= c("B11") # K2,3 Bipartite 11
Compress1$from[80:82]= c("B11")
Compress1$from[83:85]= c("B12") # K2,3 Bipartite 12
Compress1$from[86:88]= c("B12")
Compress1$from[89:91]= c("B13") # K2,3 Bipartite 13
Compress1$from[92:94]= c("B13")
Compress1$from[95:97]= c("B14") # K2,3 Bipartite 14
Compress1$from[98:100]= c("B14")
```

```
attach(Compress1)
from
Compress1$to[1:3]= c("B1") # K2,3 Bipartite 1
Compress1$to[4:6]= c("B1")
Compress1$to[7:9]= c("B2") # K2,3 Bipartite 2
Compress1$to[10:12]= c("B2")
Compress1$to[13:13]= c("B3") # K2,3 Bipartite
Between All_1
Compress1$to[14:14]= c("B4")
Compress1$to[15:15]= c("B5")
Compress1$to[16:16]= c("B3")
Compress1$to[17:17]= c("B4")
Compress1$to[18:18]= c("B5")
Compress1$to[19:21]= c("B3") # K2,3 Bipartite 3
Compress1$to[22:24]= c("B3")
Compress1$to[25:27]= c("B4") # K2,3 Bipartite 4
Compress1$to[28:30]= c("B4")
Compress1$to[31:31]= c("B8") # K2,3 Bipartite
Between All_2
Compress1$to[32:32]= c("B9")
Compress1$to[33:33]= c("B10")
Compress1$to[34:34]= c("B8")
Compress1$to[35:35]= c("B9")
Compress1$to[36:36]= c("B10")
Compress1$to[37:39]= c("B5") # K2,3 Bipartite 5
Compress1$to[40:42]= c("B5")
Compress1$to[43:45]= c("B6") # K2,3 Bipartite 6
Compress1$to[46:48]= c("B6")
Compress1$to[49:51]= c("B7") # K2,3 Bipartite 7
Compress1$to[52:54]= c("B7")
Compress1$to[55:57]= c("B8") # K2,3 Bipartite 8
Compress1$to[58:60]= c("B8")
Compress1$to[65:67]= c("B9") # K2,3 Bipartite 9
Compress1$to[68:70]= c("B9")
Compress1$to[71:73]= c("B10") # K2,3 Bipartite 10
Compress1$to[74:76]= c("B10")
Compress1$to[77:79]= c("B11") # K2,3 Bipartite 11
Compress1$to[80:81]= c("B11")
Compress1$to[82:82]= c("B12")
Compress1$to[83:85]= c("B12") # K2,3 Bipartite 12
Compress1$to[86:87]= c("B12")
Compress1$to[88:88]= c("B13")
Compress1$to[89:91]= c("B13") # K2,3 Bipartite 13
Compress1$to[92:93]= c("B13")
Compress1$to[94:94]= c("B14")
Compress1$to[95:97]= c("B14") # K2,3 Bipartite 14
Compress1$to[98:99]= c("B14")
Compress1$to[100:100]= c("B11")
attach(Compress1)
Compress1
write.csv(Compress1, file =
"MyData.csv",row.names=FALSE)
CompressBipartite1<- read.csv("MyData.csv",
header=T, as.is=T)
CompressBipartite1
igraph2 <- graph.data.frame(CompressBipartite1)
igraph2
V(igraph2)$type <- V(igraph2)$name %in%
Compress[,1]
bipartite.projection(igraph2)
bipartite_mapping(igraph2)
lc <- largest.cliques(igraph2)
lc
# Fruchterman.reingold.layout(Tree layout)
V(igraph2)$frame.color <- "white"
V(igraph2)$color = "orange"
V(igraph2)$size = 11
E(igraph2)$arrow.mode = 0
plot(igraph2, vertex.label.cex=.4,
layout=layout.fruchterman.reingold,
mark.groups=list(c("B1"),c("B2"),c("B3"),c("B4"),
c("B5"),c("B6"),c("B7"),c("B8"),c("B9"),c("B10"),
c("B11"),c("B12"),c("B13"),c("B14")),mark.col=c(
"green","red", "yellow", "blue", "orange", "cyan",
"fireBrick", "black", "brown", "magenta",
"tomato", "pink", "salmon", "purple"),main="Tree
layout (14 K2, 3 Bipartite")
legend(x=-1.5, y=-1.1, c("K2,3 Bipartite-1"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="green")
legend(x=-1.5, y=-1.2, c("K2,3 Bipartite-2"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="red")
legend(x=-1.5, y=-1.3, c("K2,3 Bipartite-3"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="yellow")
legend(x=-1.5, y=-1.4, c("."), pch=21,
col="#777777", pt.cex=2, cex=.8, bty="n", ncol=1,
pt.bg="white")
legend(x=-1.5, y=-1.5, c("."), pch=21,
col="#777777", pt.cex=2, cex=.8, bty="n", ncol=1,
pt.bg="white")
legend(x=-1.5, y=-1.6, c("K2,3 Bipartite-14"),
pch=21, col="#777777", pt.cex=2, cex=.8, bty="n",
ncol=1, pt.bg="purple")
```

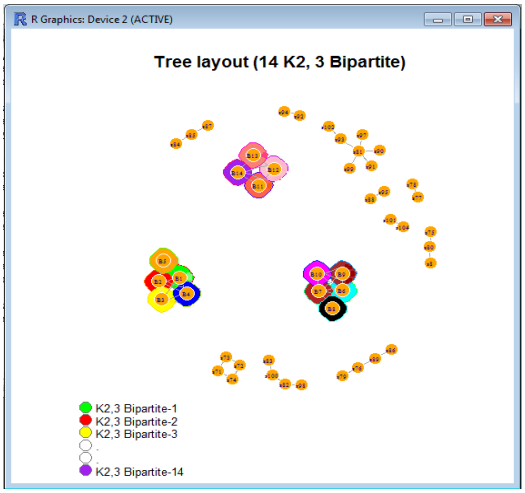

Fig. 40. Tree Layout of 14 $K_{2,3}$ (First Compress).

After compressing the $K_{2,3}$ and creating the new graph, the next step is to check if there is another $K_{2,3}$ in the graph by using the Linkcomm Package feature (loop) as shown in figure 41.

```
lc <- getLinkCommunities(CompressBipartite1,
hcmethod = "single")
```

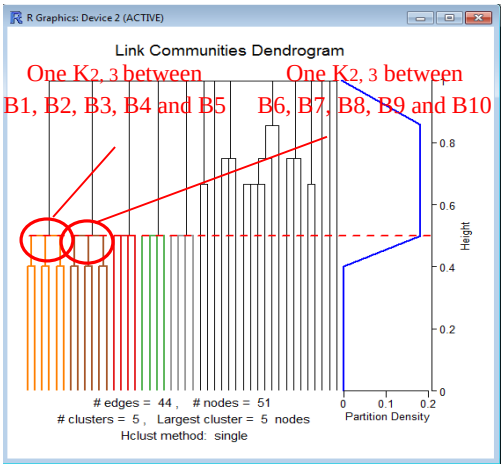

Fig. 41. Linkcomm Package- First Compress the $K_{2,3}$.

Figure 41 is not a tidy graph, because it has two $K_{2,3}$. After the $K_{2,3}$ is located, it needs to be compress to makes this graph a tidy graph by compressing this $K_{2,3}$ and creating a new graph. The following procedure is to assign all the nodes in the $K_{2,3}$ to one letter, number or word to be come all these nodes (5 node in one cluster) in the same number, letter or word. The next step is to assign all the compressing nodes in the first $K_{2,3}$ to "All" followed by a number with a color is assigned around the compressing node, for example, "All_1" with green color is assigned around the compressing node which meant this node call "All" is mean: B1, B2, B3, B4 and B5. "All_2" with red color is assigned around the compressing node which meant this node call "All" is mean: B6, B7, B8, B9 and B10. On the other hand, these $K_{2,3}$ which are not in one $K_{2,3}$ we will deal with it as normal $K_{2,3}$ (compress) such is: Tomato color is assigned around the compressing node for the fifth $K_{2,3}$ with the "B11." Pink color is assigned around the compressing node for the fifth $K_{2,3}$ with the "B12." Salmon color is assigned around the compressing node for the fifth $K_{2,3}$ with the "B13." Purple color is assigned around the compressing node for the fifth $K_{2,3}$ with the "B14" as shown in figure 42.

```
Compress2 <- data.frame(CompressBipartite1)
Compress2
attach(Compress2)
Compress2$from[1:30]= c("All_1")
Compress2$from[37:42]= c("All_1")
Compress2$from[31:36]= c("All_2")
Compress2$from[43:60]= c("All_2")
Compress2$from[65:76]= c("All_2")
attach(Compress2)
from
Compress2$to[1:30]= c("All_1")
Compress2$to[37:42]= c("All_1")
Compress2$to[31:36]= c("All_2")
Compress2$to[43:60]= c("All_2")
Compress2$to[65:76]= c("All_2")
attach(Compress2)
Compress2
write.csv(Compress2, file =
"MyData.csv",row.names=FALSE)
CompressBipartite2<- read.csv("MyData.csv", header=T,
as.is=T)
CompressBipartite2
igraph3 <- graph.data.frame(CompressBipartite2)
igraph3
V(igraph3)$type <- V(igraph3)$name %in%
Compress2[,1]
bipartite.projection(igraph3)
bipartite_mapping(igraph3)
# Fruchterman Reingold Layou(Tree Layou)
V(igraph3)$frame.color <- "white"
V(igraph3)$color = "orange"
V(igraph3)$size = 11
E(igraph2)$arrow.mode = 0
plot(igraph3, vertex.label.cex=.4,
layout=layout.fruchterman.reingold,
mark.groups=list(c("All_1"),c("All_2"),c("B11"),c("B12"
),c("B13"),c("B14")),mark.col=c("green","red","tomato",
"pink", "salmon", "purple"), main="Tree layout (14 K2, 3
Bipartite")
legend(x=-1.5, y=-1.1, c("Five K2,3 in one K2,3 (B1-
B5)"), pch=21, col="#777777", pt.cex=2, cex=.8,
bty="n", ncol=1, pt.bg="green")
legend(x=-1.5, y=-1.2, c("Five K2,3 in one K2,3 (B6-
B10)"), pch=21, col="#777777", pt.cex=2, cex=.8,
bty="n", ncol=1, pt.bg="red")
legend(x=-1.5, y=-1.3, c("K2,3 Bipartite-B11"), pch=21,
col="#777777", pt.cex=2, cex=.8, bty="n", ncol=1,
pt.bg="tomato")
legend(x=-1.5, y=-1.4, c("K2,3 Bipartite-B12"), pch=21,
col="#777777", pt.cex=2, cex=.8, bty="n", ncol=1,
pt.bg="pink")
legend(x=-1.5, y=-1.5, c("K2,3 Bipartite-B13"), pch=21,
col="#777777", pt.cex=2, cex=.8, bty="n", ncol=1,
pt.bg="salmon")
legend(x=-1.5, y=-1.6, c("K2,3 Bipartite-B14"), pch=21,
col="#777777", pt.cex=2, cex=.8, bty="n", ncol=1,
pt.bg="purple")
```

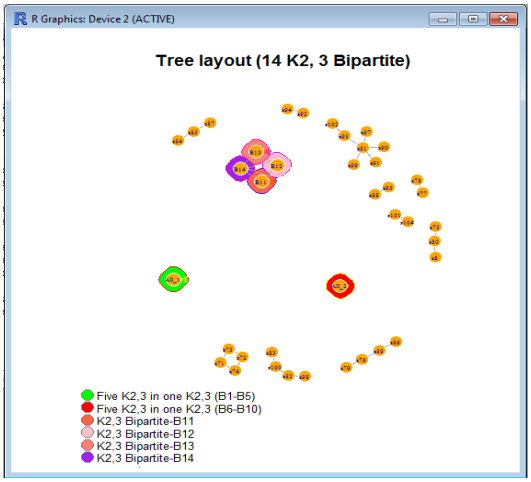

Fig. 42. Tree Layout of 14 $K_{2,3}$ (Second Compress).

After compressing the $K_{2,3}$ and creating the new graph, the next step is to check if there is another $K_{2,3}$ in the graph by using the Linkcomm Package feature again (loop) as shown in figure 43.

```
lc <- getLinkCommunities(CompressBipartite2,
hcmethod = "single")
```

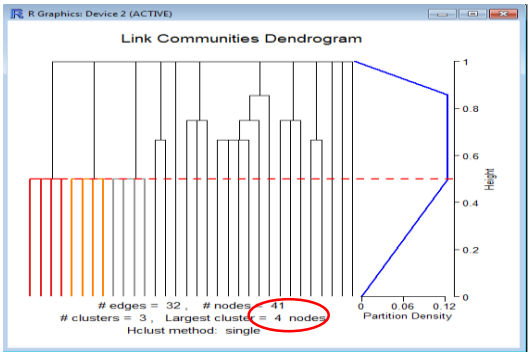

Fig. 43. Linkcomm Package- Second Compress the $K_{2,3}$.

In this dataset, there are 14 $K_{2,3}$ in the graph and some of these $K_{2,3}$ in one $K_{2,3}$. The last part of the algorimetic solutions of the hypothesis was used to"Represent $K_{2,3}$"through R. To represent these $K_{2,3}$ by R, we selected and customized one of the available layout which is Tree Layout as shown in figure 44 and 45.

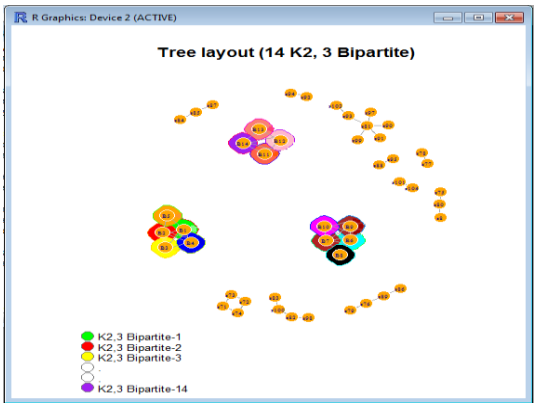

Fig. 44. First Represent Graph by Tree Layout.

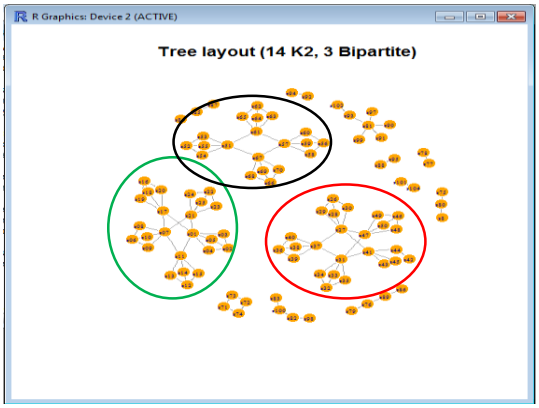

Fig. 45. Second Represent Graph by Tree Layout.

5.3.3 Big Dataset 3-Time Use (30 $K_{2,3}$ Bipartite)—The Big datasets of Time Use in excel contains 30 $K_{2,3}$ Bipartite. In this dataset, there are "40" columns with 119 rows. The main column in this dataset is "GEOACL00 Column" which is the name of the counters. It cannot use the linkcomm package at one time for the dataset because this dataset is large. Then, we will use the linkcomm package by comparing every single column in this dataset with the main column which is "GEOACL00" to search about any $K_{2,3}$. The linkcomm package is a tool that provides the generation, visualization, and analysis of link communities in networks of arbitrary size and type. Before this package or any package is used the "igraph package with it library," must be used for a Network Analysis and Visualization. Figure 48, is the first example showing the linkcomm package by comparing between the main column (GEOACL00) and the column of "Study" that has two $K_{2,3}$.

```
#linkcomm package
StudyGLC <- getLinkCommunities(Study,
hcmethod = "single")#K2,3
```

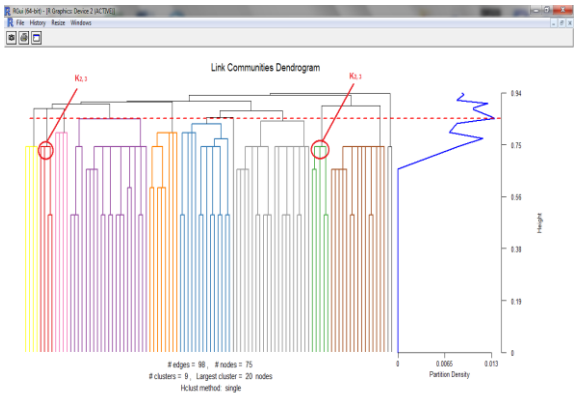

Fig. 46. $K_{2,3}$ by Linkcomm Package.

After confirming the $K_{2,3}$ bipartite is ineligible for this dataset, we will use a special layout to make sure if this graph a "tidy" graph or not by using a Random Layout as shown in figure 47.

```
# Randomly layout
V(igraph)$frame.color <- "white"
V(igraph)$color = "orange"
V(igraph)$size = 11
E(igraph)$arrow.mode = 0
plot(igraph, layout=layout.random,
main="Random Layout",vertex.label.cex=.4)
```

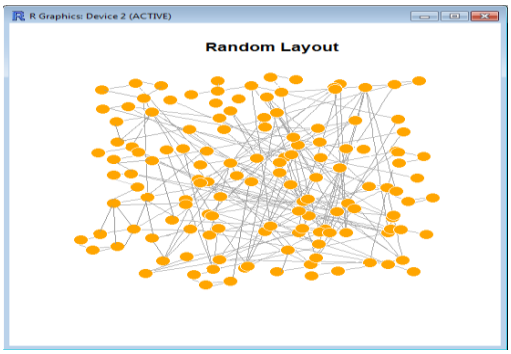

Fig. 47. Random Layout.

Figure 47 is not a tidy graph because it has 30 $K_{2,3}$. After these $K_{2,3}$, are located the next step is to makes this graph a tidy graph by compressing these $K_{2,3}$ and creating a new graph. The following procedure is to assign all the nodes in the $K_{2,3}$ to one letter or number to be come all these nodes (5 node in one cluster) in the same number or letter. The next step is to assign all the compressing nodes in the first $K_{2,3}$ to the letter B followed by a number, for example, "B1" for the compressing node for the first $K_{2,3}$. Compressing node for the second $K_{2,3}$ with the "B2." Compressing node for the third $K_{2,3}$ with the "B3" and so on. The final graph in this dataset looks like two separate graphs shown in figure 48.

```
igraph2 <- graph.data.frame(CompressBipartite1)
igraph2
V(igraph2)$type <- V(igraph1)$name %in%
CompressBipartite1[,1]
bipartite.projection(igraph2)
bipartite_mapping(igraph2)
lc <- largest.cliques(igraph2)
lc
# Fruchterman.reingold.layout(Tree layout)
V(igraph2)$frame.color <- "white"
V(igraph2)$color = "orange"
V(igraph2)$size = 6
E(igraph2)$arrow.mode = 0
plot(igraph2, vertex.label.cex=.1,
layout=layout.fruchterman.reingold, main="Tree
layout (31 K2, 3 Bipartite)")
```

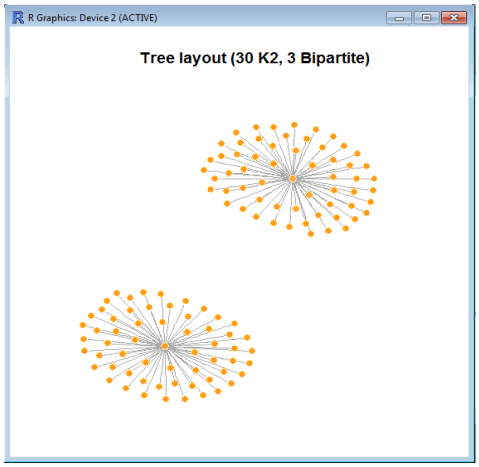

Fig. 48. Tree Layout.

In this dataset, there are 30 $K_{2,3}$ in the graph. The last part of the algorimetic solutions of the hypothesis was used to"Represent $K_{2,3}$"through R. To represent these $K_{2,3}$ by R, we selected and customized one of the available layout which is Tree Layout as shown in figure 49.

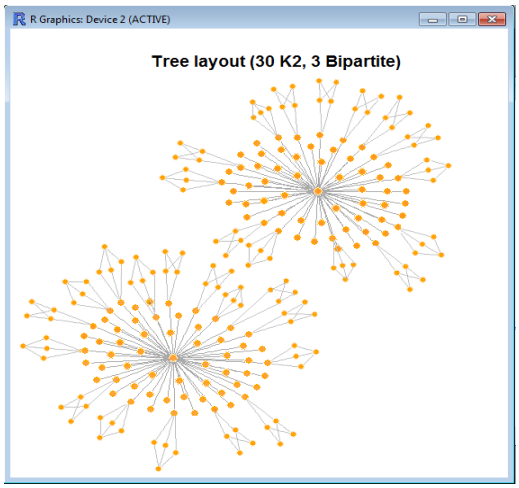

Fig. 49. Represent Graph by Tree Layout.

6. RUNTIME

The average of 100 runtime by R for these datasets show that when the datasets have two levels in $K_{2,3}$ ($K_{2,3}$ in $K_{2,3}$) the average of 100 runtime increases in comparison to small, one-level, datasets. More levels in the datasets increase 100 runtime are to a more complex algorithm (e.g. two level in $K_{2,3}$ in the dataset meant this dataset is a complex dataset) as shown in the table 7 [14].

Dataset	Average of 100 Runtime
Small Datasets (No $K_{2,3}$ Bipartite)	2.65 secs
Small Datasets (One $K_{2,3}$ Bipartite)	2.628 secs
Small Datasets (Two $K_{2,3}$ Bipartite)	2.684 secs
Small Datasets (Five $K_{2,3}$ Bipartite in One $K_{2,3}$)	3.04 secs
Big Datasets - AnAge Dataset (No $K_{2,3}$ Bipartite)	3.90 secs
Big Datasets - Media Dataset (14 $K_{2,3}$ Bipartite)	4.401 secs
Big Datasets - Time Use Dataset (30 $K_{2,3}$ Bipartite)	5.639 secs

Table 7: Average of 100 Runtime for all the Dataset by R.

7. BIG-O(BIGOH) FOR TIME COMPLEXITY OF ALGORITHMS

Big-O [13] is an expression for the execution time of a program (how long the program will take). There is theorem that we will use it in this part [25]:

- **Theorem: O ((e+n)n)**
  log (e+n) + log(n)

Notes:

- **n =** number of nodes
- **e** = number edges
- **RAM** = 4.00 GB
- **CPU** = 2.40 GHZ
- **System type** = 64-bit

Comparison between the outcomes of the average of 100 runtime coulmen and the Big-O(BigOh) coulmen for the Time Complexity of Algorithms show us that these columns are equal which meant that this theorem is a correct as showen in Table 8 [11]:

Dataset	n	e	Average of 100 Runtime	Gig-O
Small Datasets 1 (No $K_{2,3}$ Bipartite)	15	12	2.65 secs	2.60 secs
Small Datasets 2 (One $K_{2,3}$ Bipartite)	15	13	2.628 secs	2.62 secs
Small Datasets 3 (Two $K_{2,3}$ Bipartite)	15	17	2.684 secs	2.68 secs
Small Datasets 4 (Five $K_{2,3}$ Bipartite)	25	36	3.04 secs	3.18 secs
Big Datasets 1-AnAge (No $K_{2,3}$ Bipartite)	197	150	3.90 secs	4.83 secs
Big Datasets 2-Media (14 $K_{2,3}$ Bipartite)	124	103	4.401 secs	4.44 secs
Big Datasets 3-Time Use (30 $K_{2,3}$ Bipartite)	238	505	5.639 secs	5.33 secs

Table 8: Big-O(BigOh) for Time Complexity of Algorithms.

8. CONCLUSION

- **It is a feasible to use $K_{2,3}$ as a key vector to prune a complicated graph.**
  - Feasible is means:
    - Dataset is doable (is works).
    - Time is tolerable
- **The mathematical theorem (O ((e+n)n)) is useful for the algorithm.**
  - Useful means:
    - The average of 100 runtime results were identical to the Big-O results.

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my deepest appreciation to The Saudi Arabian Cultural Mission (SACM) for giving me their valuable suggestions and support throughout my research.

9. REFERENCES

[1] A. Nocaj., M.Ortmann.,&U.Brandes,'Untangling hairballs. From3to14DegreesofSeparation',[Online].Available:https ://www.unikonstanz.de/mmsp/pubsys/publishedFiles/No OrBr14.pdf. [Online]. [Accessed: 11- Sept - 2015].

[2] A. Ryan., &H. Mark, 'Networksis: a package to simulate bipartite graphs with fixed marginals through sequential importance sampling', Journal of Statistical Software, vol. 24, pp. 1-21, 2008.

[3] A.T. Kalinka, 'The generation, visualization, and analysis of link communities in arbitrary networks with the R package linkcomm'. Vienna, Austria. pp.1-16, 2014.

[4] B. Bai., W. Chen., K. Letaief., &Z. Cao, 'Diversity-Multiplexing Tradeoff in OFDMA Systems: An $\mathcal{H}$-Matching Approach', IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, vol. 10, pp. 3675-3687, 2011.

[5] B. Carter, 'network: A package for managing relational data in r', Journal of Statistical Software, vol. 24, pp. 1-36, 2008.

[6] B. Ghosh. Ranjan., S. Banerjee., S. Dey., S. Ganguli & Sarkar S, 'Off-Line signature verification system using weighted complete bipartite graph', 2nd International Conference on Business and Information Management (ICBIM), pp. 109-113, 2014.

[7] C. Dormann, G. Bernd, &F. Jochen, 'Introducing the bipartite package: analysing ecological networks', vol. 8/2, pp. 7-11, 2008.

[8] Carsten F. Dormann., J. Fruend., B. Gruber., with additional code from M. Devoto., J. Iriondo., R. Strauss., & D. Vazquez, also based on C-code developed by N. Bluethgen., A. Clauset/Rouven Strauss & M. Rodriguez-Girones, 'Package 'bipartite', pp. 1- 154, 2014.

[9] C. Dunne &S. Ben, 'Motif simplification: improving network visualization readability with fan, connector, and clique glyphs', Changing Perspectives, Paris, France, pp. 3247-3256, 2013

[10] D. Koutra, 'Exploring and making sense of large graphs' Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, pp. 1-231, 2015

[11] D. Joyner., M. Van Nguyen, N. Cohen, 'Algorithmic Graph Theory', pp. 1-163, (2009-2010)..

[12] E. Hartung, 'The linear and cyclic wirelength of complete bipartite graphs', pp. 1-20, 2004.

[13] E. Tardos., and J. Kleinberg, 'Algorithm Design'. Pp. 1-10, 2006.

[14] Heymann, S., Grand, L., &B´en´edicte, 'Visual analysis of complex networks for business intelligence with gephi', 17th International Conference on Information Visualisation, vol. 13, pp. 307-312, 2013.

[15] G. Csardi, 'Package igraph', Network Analysis and Visualization, pp. 1-462, 2015.

[16] J. Gentry., R. Gentleman &W. Huber, 'How to plot a graph using rgraphviz', pp. 1-25, 2015.

[17] Ken, Cherven. (2015). *Mastering Gephi Network Visualization*. Birmingham, UK: Packt Publishing Ltd

[18] K. Etemad., S. Carpendaleya &F. Samavatiz, 'Spirograph inspired visualization of ecological networks', pp. 81-91, 2014.

[19] K. O'HAIR, 'The geometric structure of spanning trees and applications to multiobjective optimization' pp. 1-41, 2009

[20] K, Rosen, 'CS 137 - Graph theory - lectures 4-5', Discrete Mathematics and its Applications, 5th ed, chapters. 8.7, 8.8, 2012.

[21] Martin, G, Everett., Jeffrey, C, Hohnson., Stephen Borgatti. (2013). *Analyzing Social Networks*. Los Angels, CA. London, UK. New Delhi, India. Washington, DC: MPG Printgroup

[22] Maris, L. Rizzo. (2008). Statistical Computing with R. Bowling Green, OH: Bowling Green University.

[23] M. Solomon, 'Working with bipartite/affiliation network data in R. Generating Labels for Supervised Text Classification using on plotting numeric data by groups', 2012.[Online].Available:https://solomonmessing.wordpress.com/2012/09/30/working-with-bipartiteaffiliation-network-data-in-r/. [Accessed: 11- Aug - 2015].

[24] M. Gori., M. Maggini., & L. Sarti, 'Exact and approximate graph matching using randomly walks', IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, vol. 27, pp. 1100-1111, 2005.

[25] M. Parker., C. Lewis, 'Why is Big-O Analysis Hard?', 2013. [Online]. Available: http://dx.doi.org/10.1145/2526968.2526996. [Accessed: 15- February - 2016].

[26] O. Katherine, 'Static and dynamic network visualization WithR',2015.[Online].Available: https://rpubs.com/kateto/netviz. [Accessed: 11- Aug - 2015].
[27] Richard Brath, David Jonker. (2015). *Graph Analysis and Visualization*. Indianapolis. IN: John Wiley & Sons, Inc.
[28] S. Lehmann., M. Schwartz &K. Hansen3,'Bi-clique Communities, 1Center for Complex Network Research and Department of Physics', Northeastern University, Boston, pp. 1-10, 2013
[29] T. Awal., M. Rahman, 'A linear algorithm for resource four-partition four-connected planar graphs', 6th International Conference on Electrical and Computer Engineering ICECE. , pp. 526- 529, 2010.
[30] 'The Open Graph Viz Platform', [Online]. Available: https://gephi.org/.[Online]. [Accessed: 01- January - 2016].
[31] Y. Egawa &M. Furuya, 'Forbidden triples containing a complete graph and a complete bipartite graph of small order',Graphs and Combinatorics, pp. 1149–1162, 2014.
[32] Y. Hu., H. Jin., S. Lin & H. Li, 'A study of pseudo-automorphism groupabout the cn',Second International Conference on Information Technology and Computer Science, vol. 10, pp. 332-335, 2010.

Khalid H. Alnafisah is completing a Master of Science degree in Computer and Information Science from Gannon University, PA, with Expected Graduation May 2016. He currently holds a Bachelor of Science degree in Computer Science from Grambling State University, LA, and High diploma in Communication and Computer Networks from the Kingdom of Saudi Arabia (2006). He has some publications including 4 patents from the U.S.A patent office.