Application of Graph Theory in Fingerprint Classification

Yoel Susanto 13517014
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13517014@std.stei.itb.ac.id

***Abstract*—Fingerprint is the pattern or impression found on the human fingertip. Fingerprint is unique to every person and it last a person's whole life. Such features caused the widespread use of fingerprint for identification purpose. The need for personal identification is ever growing as technological advances open various occasion where we would be required to identify ourselves. Ranging from unlocking our phones, registering our attendance, withdrawing money from ATM machines, voting for election, identifying patient with mental difficulty or unknown deceased, and identifying criminal suspect. There are numerous occasions where we would need to compare a fingerprint input with the sheer number of fingerprints of the population we have in the database. Simply matching the input with each fingerprint in the database would need tremendous amount of computing power and still require a lot of time. The better approach is to classify the input fingerprint and only do the matching with the fingerprints which belongs to the same class of the input finger print. This paper will discuss on method of classifying fingerprint using graph theory.**

***Keywords*—Directional image, Graph application, Fingerprint classification, Fingerprint matching.**

I. Introduction

Personal identification has become more and more frequent as technological advances keep developing. Fingerprints are so widely used as biometric characteristics for personal identification. However, personal identification using fingerprint doesn't come without challenges. The amount of resources needed for doing comparison of an input fingerprint with every fingerprint on the database is too big to accommodate. To solve such situation, researchers proposed a solution called fingerprint classification.

The goal to classifying fingerprint is to reduce the number of matching needed to be performed upon the input fingerprint and the fingerprints stored in the database. There are many approaches to classifying fingerprints. Some proposed on calculating the number of singularity points such as the core and the delta points that are present in the fingerprints. While this method works well on high quality images of fingerprints, it is not reliable when the image obtained from the fingerprint is significantly corrupted. In dealing with such problem, researchers proposed algorithm that rely only partially on calculation of singularity point and in addition to that they try to calculate the orientation field of the fingerprints. This paper will focus on the proposed method of fingerprint classification by computing the orientation field of a fingerprint as well as on the use of graph theory in the process.

II. Background On Graphs

A. Definition

Graphs are discrete structures consisting of vertices and edges that connect these vertices. There are different kinds of graphs, depending on whether edges have directions, whether multiple edges can connect the same pair of vertices, and whether loops are allowed.[2]

Formally, *graphs* are denoted as

$$G = (V, E)$$

where V is a nonempty set of *vertices, nodes, or points* and E is a set of *edges, lines or arcs*. The vertices and edges inside a graph can represent different kind of objects and relations, their meanings differ significantly depending on the context where the graph is being used to represent information.

Vertices on graph are usually labelled using alphabet or numbers or the combination of both while the edges are usually labelled using *e* followed by a number. subscript Vertices can have multiple or no edges associated with it, while edges must have either one or two vertices associated with it. Edges are commonly denoted as

$$e = (u, v)$$

where u and v are the vertices connected by e.

Generally, graphs are visualized using points to represent vertices and line segments to represent edges.

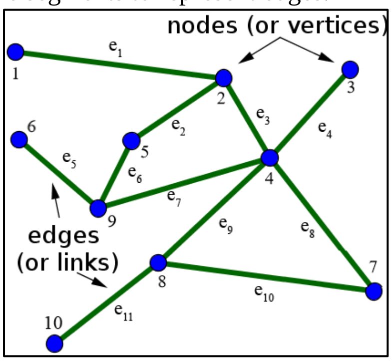

Fig 1. A graph consisting of 10 vertices and 11 edges $V = \{1, 2, ..., 10\}$, $E = \{e_1, e_2, ..., e_{11}\}$

https://mathinsight.org/media/image/image/small_undirected_network_labeled.png (accessed December 8, 2018)

B. Types of Graph

There are many ways to classify graphs. Based on the existence of parallel edges, graphs can be classified into two kinds:

1. **Simple Graph**
   A simple graph is a graph with no loop or parallel edges connecting a pair of vertices. Edges in simple graph are concerned with the order of the vertices they are connected to. We can safely say that edge (*u*, *v*) is the same as edge (*v*, *u*).

2. **Multigraph**
   Multigraph are graphs consisting of parallel edges. In addition, when a vertex in a graph has edge connecting to itself, the graph is classified as **pseudograph**.

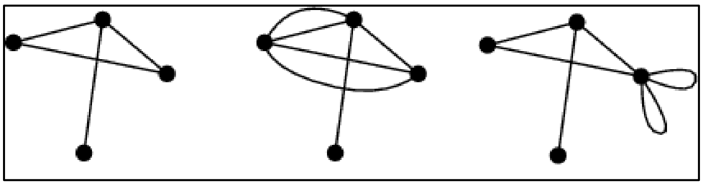

Fig 2. Simple graph(left), multigraph(middle) and pseudograph(right)

http://mathworld.wolfram.com/images/eps-gif/SimpleGraph_950.gif (accessed December 8, 2018)

Based on the whether the edge of a graph is associated with ordered pair of vertices or not, we classify graph into two kinds:

1. **Undirected Graph**
   Edges in directed graph are not associated with any order of the vertices they connected. Edge (*u*, *v)* is the same as edge (*v*, *u*).
2. **Directed Graph**
   On the other hand, directed graphs are graph with edges that have direction associated with it. In this case we say that edge (*u*, *v) start* at *u* and *end* at *v*. This means that edge (*u*, *v)* is not the same as edge (*v, u)*.

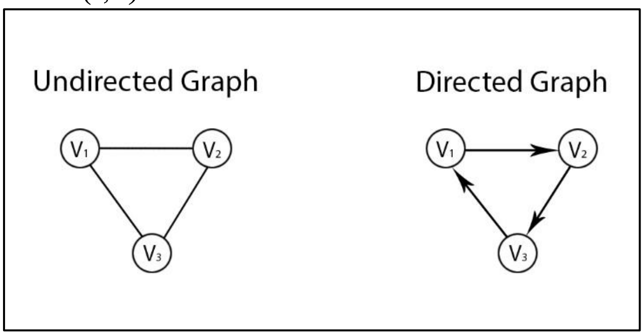

Fig 3. Undirected and directed graph

https://www.e-education.psu.edu/geog597i_02/sites/www.e-education.psu.edu.geog597i_02/files/Lesson8/Geog597i_Lesson8_directedgraph.jpg (accessed December 8, 2018)

C. *Graph Terminology*

1. Adjacent
   If two vertices *u* and *v* in a graph *G* is connected by an edge, *u* and *v* are adjacent in *G*.
2. Incident
   An edge is incident with both of the vertex it connected.
3. Isolated Vertex
   The isolated vertex in a graph is a vertex which doesn't have any edge incident with it.
4. Null Graph
   Null graph is a graph which doesn't contain any edges. Nonetheless the null graph may contain any number of vertices.
5. Degree
   Degree of a vertex inside a graph is the sum of edges associated to it.
6. Path
   Path in a graph is a sequence of edges that travels from one vertex to the other vertices.
7. Circuit
   Circuit is path that begins and ends at the same vertex.
8. Connected
   Two vertices *a* and *b* inside a graph are said to be connected if there exist a path that begins at *a* and ends at *b*.
9. Subgraph
   A subgraph *H* of graph *G* which contain a subset of vertices that G has and also contain a subset of the edges that G has.
10. Spanning Subgraph
    A subgraph of a graph which contains all the vertices inside the original graph is called the spanning subgraph.
11. Cut-Set
    A cut-set of a graph is a set of edges which would cause a graph not to be connected if it is removed.
12. Weighted Graph
    A graph *G* is called as weighted graph if edges inside graph *G* has a value assigned to it instead of just lines.
13. Special Graph
    There are several kinds of special graph such as: complete graph, regular graph and bipartite graph.

D. *Representing Graphs*

It is easy for human to represent graph as depicted in pictures using dots and lines, but it is not such in the case of representing graph in computers. As computer store data using ones and zeros, we cannot simply draw a graph into graph. We need to a systematic and reliable way of representing graph such that the data could be meaningful. There are several ways we could represent graph:

1. Adjacency Matrix
   When we use adjacency matrix to represent graph, we define a matrix *A* the size of *N x N* where *N* is the number of nodes inside a graph. Suppose a_{ij} is an element of A, a_{ij} is 1 if there exist an edge connecting v_i and v_j. Otherwise, a_{ij} will equal to 0.

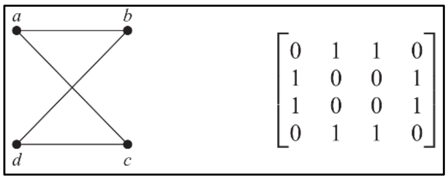

Fig 4. Example of a graph and its corresponding adjacency matrix

Discrete Mathematics and Its Applications, Seventh Edition, Kenneth H Rosen, Page 669

2. Incidence Matrix
   Incidence matrix focus on vertex and which edges is it incident with. If we have a graph *G = (V, E)* consisting of vertices $v_1, v_2, \ldots, v_n$ and edges $e_1, e_2, \ldots, e_m$, then the incidence matrix will have a size of *n x m*. Suppose m_{ij} is an element of M, m_{ij} is 1 if e_j is incident with v_i and 0 otherwise.

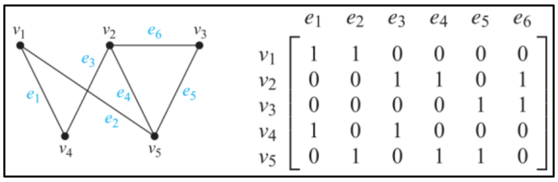

Fig 5. Example of a graph and its corresponding incidence matrix

Discrete Mathematics and Its Applications, Seventh Edition, Kenneth H Rosen, Page 669

III. Background On Biometrics and Fingerprint

A. Biometrics

Biometric characteristic is any human physiological and or behavioral characteristic. People sought to look for good biometric characteristic for the purpose of personal identification. Such characteristic should be universal, distinctive, permanent and collectable. In addition to that, for a practical biometric system, some additional criteria must be included into consideration, such as performance, acceptability and circumvention. There are a lot of biometric systems proposed for personal identification such as fingerprint scan, face contour identification, iris scan, voice recognition, retinal scan, hand geometry scan and palmprints scan. Generally, the tradeoff between the different kinds of biometric system are accuracy and cost. Some system doesn't cost much but it has poor accuracy while the other has great accuracy but cost much. The goal is to find the sweet spot where the cost to develop the system is affordable while the accuracy of the system is reliable. One of such system is fingerprint which we will be focusing on.

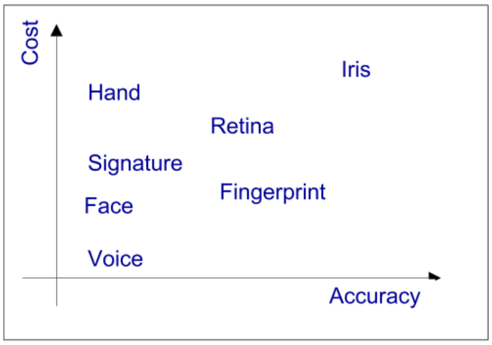

Fig 6. Trade-off between accuracy and costs for various biometrics system

STRUCTURAL AND GRAPH-BASED METHODS FOR AUTOMATIC FINGERPRINT CLASSIFICATION, Alessandra Serrau, 2007, page 9

B. Features of Fingerprint

Fingerprint recognition system rely on two assumptions: Invariance and Singularity. Invariance means a person's fingerprint will never change along the life. Singularity means the fingerprint someone has is unique and there are no two same fingerprints in the world. the unique features that fingerprint has. Finger print features is divided into two:

1. Local Features (Micro characteristic)
   Local features are features used for positive identification, that is the process of identification when someone claims an identity and we perform the check if the claimed identity matches his or hers. Features used to do such identification must be very unique to every person. Such features in fingerprint are known as minutiae points. Minutiae points are points where ridge lines bifurcate or terminate.

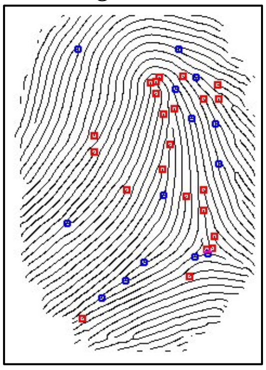

Fig 7. Minutiae points on fingerprints. Blue is bifurcation, Red is termination

https://www.researchgate.net/profile/Sunny_Sudiro/publication/303407988/figure/fig1/AS:364747934453760@1463974092153/Fingerprint-image-with-minutiae-pooints-extacted-and-their-vector-template-red-points_W640.jpg (accessed December 8, 2018)

2. Global Features (Macro characteristic)
   Global features are feature commonly found on most fingerprints, although these features also differ between fingerprints, the differences are sufficient to be used for identification, instead they are generally used for classifying fingerprints. These features include:
   - Core
     Core is the approximate center of the fingerprints.
   - Pattern Area
     Pattern area in fingerprints is the area where we are interested in, mainly because it is the area which contain the other macro characteristic such as delta, core and ridges.
   - Type Lines
     Type lines are the two lines that surround the pattern area. They usually start in a parallel manner.
   - Delta
     The delta is that point on a ridge at or in front of and nearest the center of the divergence of

the type lines. [6]

- Ridge Count

  Ridge Count is the number of ridges that exist between the delta and the core of the fingerprint. Usually it is counted by drawing an imaginary line that begins at the delta and ends at the core. The number of ridges that this line touches will be the ridge count of the fingerprint.

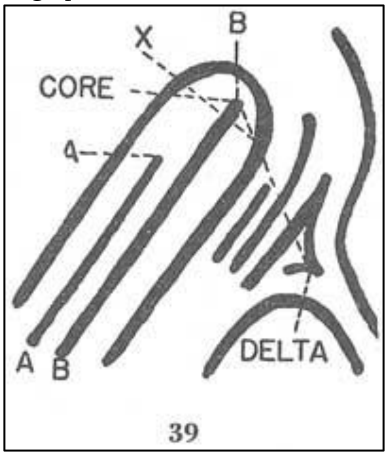

Fig 8. The core and delta in a fingerprint

https://www.gutenberg.org/files/19022/19022-h/images/fig039.jpg (accessed December 8, 2018)

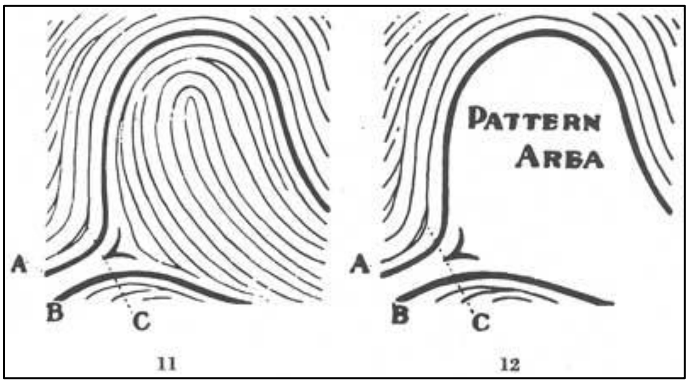

Fig 9. Type lines (A and B) and pattern area in a fingerprint

https://www.gutenberg.org/files/19022/19022-h/images/fig011-012.jpg (accessed December 8, 2018)

III. Computation and Segmentation of Fingerprint Directional Image

Directional image of a fingerprint is a matrix which shows the direction of the ridges in the corresponding fingerprint image. There are various proposed methods on computing the directional image of a fingerprint. The method used in this paper produces a directional image which quantified into four different direction. The reason for choosing four direction is the consideration of additional complexity when forming graph associated to the directional image and additional time needed on the process of matching. Since the purpose of the whole procedure is to classify the fingerprint, the accuracy loss compared to using directional image with more direction is not significant.

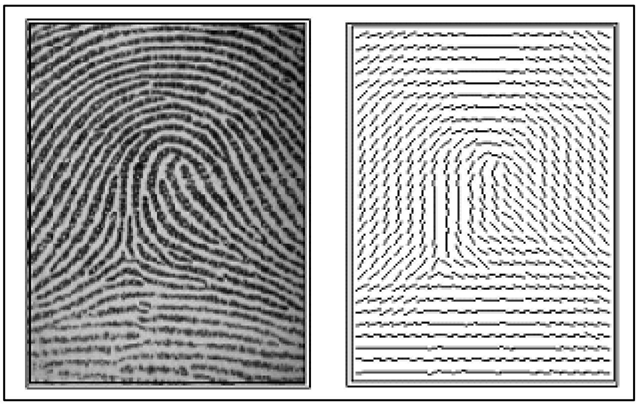

Fig 10. Example of fingerprint(left) and its corresponding directional image(right)

http://www.biometrika.it/eng/images/dirimg.gif (accessed December 8, 2018)

After obtaining the directional image of fingerprint, to reduce noise from the original image, we compute the dominant direction in every 4x4 blocks. This resulted into a block directional image. The block directional image is then further processed by segmenting or partitioning the elements of the matrix into homogenous regions depending on the four directions. Each region in the resulting image contain elements with similar direction and these regions will be used in computing the graph to describe the potential class the fingerprint belongs to.

Fig 11. Regions of similarly directed elements of fingerprint ridges

Automatic Fingerprint Classification using Graph Theory, Tarjoman and Zarei, 2008, page 2

IV. Constructing Related Graph Based on Segmented Block Directional Image

After transforming the image of a fingerprint into segmented block directional image, we have obtained a good base for graph construction. While it is very hard to construct graph out of plain image of fingerprint, it is relatively easy to construct graph out of segmented block directional image. There is several information that we can embed into the related graph of a fingerprint, such as: the center of gravity of the homogenous region, the direction of a particular region, the area of the region, the phase difference between the direction association with the regions and the distance between the center of gravity of each region.

The graph used in the method discussed in this paper will attribute node to a corresponding region and edges will represent the common perimeter of two adjacent region. The node will have weight associated with it according to the area of the corresponding region. The weight of each node is formulated as:

$$W(N_i) = Area(R_i), i = 1, 2, \ldots, n$$

Where n is the number of regions in the image while N_i and R_i are the node of the graph and region of the segmented block directional image respectively. The weight of edges is slightly more complicated and it is a function of three functions:

1. Adj-p is the length of the boundary separating two homogenous regions.
2. Node-d is the distance between the center of gravity between the two homogenous regions.
3. Diff-v is the phase difference of the direction between the two homogenous regions.

The weight of the edge is formulated as:

$$W(E_i) = Adj\text{-}p(u, v)\ x\ Node\text{-}d(u, v)\ x\ Diff\text{-}v(u, v)$$

Where *E* is an edge connecting node *u* and *v*.

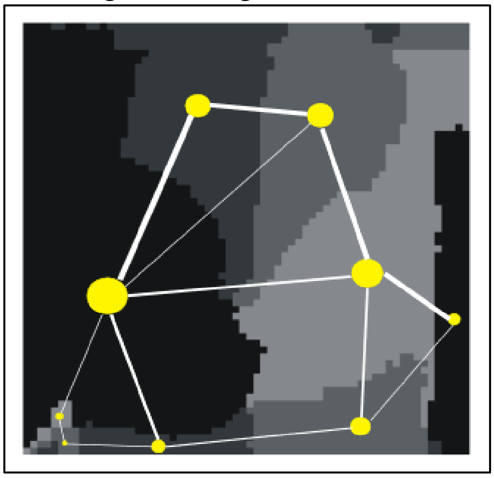

Fig 12. Graph of a segmented block directional image

Automatic Fingerprint Classification using Graph Theory, Tarjoman and Zarei, 2008, page 2

In the image shown above, the nodes are located at the center of gravity in the corresponding region. The size of each node corresponds with its weight. The edges in the graph connect neighboring regions and the thickness of the of line representing corresponds to its weight.

V. Constructing the Super Graph Related to the Fingerprint

To further reduce noise, the related graph obtained from segmented block directional image is further processed into super graph. Beside reducing noise, the super graph also reduces the complexity when comparing the input graph and the stored graph. Related graph of each fingerprint may have different number of nodes and therefore we may need to compute the subgraph of a graph to obtain a graph having the same number of nodes and edges. To solve this problem, we compute the super graph corresponding to each related graph. The super graph summaries the information contained inside the related graph into a form which can be easily processed. Each node in the super graph represents the combined regions of similar direction. The coordinate of these nodes is the mean of the center of gravity of nodes with similar direction. The weight of the node (Sn_k) is the sum of the weight of the nodes which are formed from regions with similar direction, formulated as:

$$W(Sn_k) = \sum_{i=1}^{n} Area(R_i)$$

Where k is between 1 and 4 (inclusive) and n is the number of nodes having similar direction and R_i is the region having similar direction. The weight of each edge in the super graph (Se_k) is equal to the distance between the nodes of the super graph it is connecting and the sum of the adjacent perimeter of two regions which has different directions, formulated as:

$$W(Se_k) = Distance(Sn_i, Sn_j) + \sum Adj\text{-}p(R_s, R_t)$$

The use of directional image which are segmented into four directions results in every super graph having only four nodes. Such characteristic of the super graph makes it easy to match fingerprint based on their super graph.

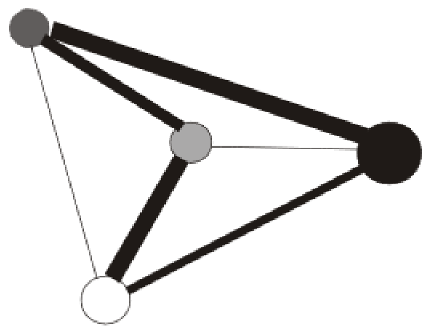

Fig 13. Super graph corresponding to the related graph on Fig. 12

Automatic Fingerprint Classification using Graph Theory, Tarjoman and Zarei, 2008, page 3

VI. Matching Super Graphs and Classification

After obtaining the related super graph, we ready for classifying the input fingerprint. Classifying is started by picking one of the fingerprints in each class of fingerprint as model fingerprint. We will compute the related super graph for each of the model fingerprint. After obtaining the related super graph of every model fingerprint, we will compare them to the related super graph from the input fingerprint by using a cost function. The cost function calculates the sum of difference between nodes and edges of two related super graphs. The cost function is formulated as:

$$Cost(u, v) = \sum_i (W(Sn_i) - W^*(Sn_i))\ x \sum_j \left(W(Se_j) - W^*(Se_j)\right)$$

Where *u* is the related super graph obtained from the input fingerprint and *v* is the related super graph of a model fingerprint. $W(Sn_i)$ and $W^*(Sn_i)$ are the node weight of super graph of the input and model fingerprint while $W(Se_i)$ and $W^*(Se_i)$ are the edge weight of super graph of the input and model fingerprint. The minimum result obtained from computing the cost function with each model fingerprint will then be used to classify the input fingerprint.

VII. CONCLUSION

There are a lot of areas where graph theory has helped in modeling various problems. We have shown how some method of fingerprint classification can benefit from graph theory. Implementation of graph theory on fingerprint classification has helped reduce the noise of fingerprint image and also the complexity of matching one fingerprint and the other. This translates into a faster process of personal identification and thus open up new opportunities where biometric identification using fingerprint would otherwise have not been possible. In the end, the author believes that there are many other opportunities where graph theory could be implemented to help solve problems.

VII. ACKNOWLEDGMENT

First, the author would like to thank God for giving the author the ability to finish the paper. The author would also like to thank Ms. Harlili M.Sc. as the lecturer of Discrete Mathematics for the guidance given for this semester. In addition, the author would also like to thank parents, family and friends who have supported the author to finish the paper.

REFERENCES

[1] M. Tarjoman and S. Zarei, "Automatic Fingerprint Classification using Graph Theory", World Academy of Science, Engineering and Technology, International Journal of Computer and Information Engineering, vol.2, no.11, 2008.
[2] K. H. Rosen, *Discrete Mathematics and Its Applications*, 7th Edition, 2013.
[3] A. Serrau, "Structural and Graph-Based Methods for Automatic Fingerprint Classification" (PhD Thesis), 2008.
[4] A. K. Jain, A. Ross, S. Prabhakar, "An Introduction to Biometric Recognition", IEEE Transactions on Circuits and Systems for Video Technology, Special Issue on Image- and Video-Based Biometrics, vol.14, no.1, January 2004.
[5] R. Sharma, N. Mishra, S. K. Yadav, "Fingerprint Recognition System and Tehniques: A Survey", International Journal of Scientific & Engineering Research, Volume 4, Issue 6, June-2013.
[6] J. E. Hoover, "The Science of Fingerprints: Classification and Uses", United States Department of Justice, Federal Bureau of Investigation, August 2006.
[7] M. Rinaldi, *Matematika Diskrit*, 3rd Edition, 2006.
[8] X. Jiang, "Fingerprint Classification", Encyclopedia of Biometrics, 2009.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 9 Desember 2018

Yoel Susanto - 13517014