Research Review
DOI: https://doi.org/10.52868/RR/2022-3-1-3
RRJ 03 (01), 597–603 (2022)
Received 14 Nov 2021 | **Revised** 31 Dec 2021 | **Accepted** 18 Jan 2022 | **Online Available** 14 Feb 2022

Research Review
MEERP

OPEN ACCESS
ISSN (O) 2693-5007

ARTICLE

Comparison of Finding the Location of the Nearest Health Facility Based On Knn-Voronoi Diagram

Sudaryanto[1] | Slamet Sudaryanto N[1*]

Abstract
Searching for an object is the implementation of one type of query that is often applied to spatial data. A common application is to find K Nearest Neighbor (KNN) of a given number of query objects. With this network, the distance between object locations depends on their network connectivity and it is computationally expensive to calculate the distance (e.g., shortest path) between objects. The term location refers to the position of a point location relative to a geometric subdivision or a discontinuous set of geometric objects. The best-known example is the point location problem, where a division of space into separate regions is given, and the problem is identifying which regions contain a particular query point. This problem is widely used in fields such as computer graphics, geographic information systems, and robotics. Point location is also used as a method for proximity search, when applied in conjunction with a voronoi diagram. In this paper, we will discuss the method of identifying the location of referral health facilities (faskes) such as hospitals, polyclinics, and the nearest health center. In this paper, we compare a new approach that is Voronoi Continuous K Nearest Neighbor (VCKNN) to efficiently locate the nearest referral health facility and evaluate KNN queries in a spatial network database using a first-order Voronoi diagram. This approach is based on partitioning large networks into small Voronoi regions, and then pre-computing distances both within and across regions. Our empirical experiments with multiple health facility location searches show that our proposed solution outperforms approaches based on online distance calculations by up to an order of magnitude, and provides a factor of four increase in filter step selectivity compared to index-based approaches. This study is shown by comparing the application of several search methods, especially for Voronoi diagram-based searches.There is a method used for comparison, namely Kd-Trees, but the resulting performance is still not satisfactory. Another method proposed is the Voronoi Continuous K Nearest Neighbor (VCKNN) algorithm which uses Voronoi diagrams to help locate nodes as objects in spatial data.

Key words: KNN, Faskes, Voronoi Diagram, KD-Trees, VCKNN

1 | INTRODUCTION

Many researchers have focused on the nearest K (KNN) query problem in spatial databases. This type of query is often used in Geographic Information Systems and is defined as: given a set of spatial objects (or points of interest), and the query point, find the K objects closest to the query. An example of a KNN query is a query initiated by a GPS device in a vehicle to find the 5 closest restaurants to the vehicle. With a spatial network database (SNDB), objects are constrained to move on a predefined path (for example, a path) defined by the underlying network. This means that the shortest network distance (e.g., shortest path, shortest time) between objects (e.g., vehicles and restaurants) depends on network connectivity rather than object location. The planar point location problem is one of the most basic query problems in computational geometry. Consider a planar

[1]Faculty of Computer Science, Dian Nuswantoro University Semarang Jl. Imam Bonjol No. 2015-207, Semarang City 50131.
Address correspondence to: Slamet Sudaryanto N, Faculty of Computer Science, Dian Nuswantoro University Semarang Jl. Imam Bonjol No. 2015-207, Semarang City 50131,

straight line graph S. This is an undirected graph, drawn on a plane, whose edges are straight line segments that have paired interiors. The s edge divides the plane into (possibly infinite) polygonal regions, called surfaces. Subsequently, such structures will be referred to as polygonal subdivisions. Along, we let n denote the combinatorial complexity S, that is, the total number of vertices, edges and surfaces (The s edge divides the plane into (possibly infinite) polygonal regions, called surfaces. Subsequently, such structures will be referred to as polygonal subdivisions. Along, we let n denote the combinatorial complexity S, that is, the total number of vertices, edges and surfaces (The s edge divides the plane into (possibly infinite) polygonal regions, called surfaces. Subsequently, such structures will be referred to as polygonal subdivisions. Along, we let n denote the combinatorial complexity S, that is, the total number of vertices, edges and surfaces (Edelsbrunner, 1987).

The point location problem is to preprocess the polygonal subdivisions of S in the plane into a data structure so that, with a query point q, the polygonal plane of the subdivision containing q can be reported quickly. This problem is a natural generalization of the binary search problem in 1-dimensional space, where the subdivision fields correspond to the intervals between 1-dimensional key values. By analogy to the 1-dimensional case, the goal is to preprocess the subdivision into a data structure of size O(n) so that point location questions can be answered in O(log n) time.

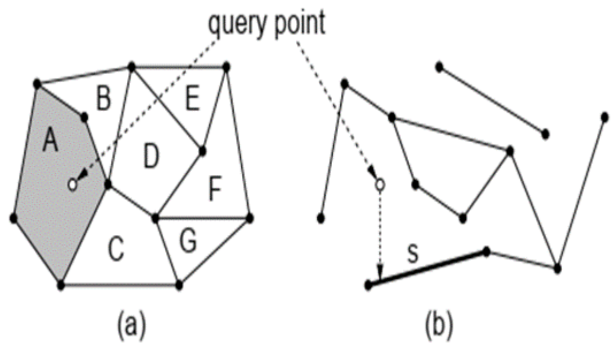

Fig. 1: *Illustration of (a) search point location and(b) vertical beam shooting query*

Given the query point q, the problem is to determine the line segment of S that lies vertically below q. In Figure 1(b), the line segment s will be reported as a crossed or crossed region or plane.A voronoi diagram is a method for dividing a region or space between a set of points that defines an area for each. The main property of this region is that each area or region represents a space in which the points are the nearest neighbors. Thus, each point in the region has been paired with the closest site (Berg et al, 2008).This study proposes a new method based on a discrete version of an adaptive voronoi t diagram that allows to divide a two-dimensional space into zones of a certain size, taking into account the position and weight of each health care ratio. The method that has been developed in finding location points on a one-dimensional plane is Kd-Trees, but the resulting performance is still not satisfactory. Approach *Continuous K Nearest Neighbor* (CKNN) has also attracted the interest of several researchers. To find the split nodes, all existing CKNN approaches divide the query path into segments, find the KNN results for the two end nodes of each segment, and then, for each segment, find the split nodes. One road segment starts at an intersection and ends at another intersection. For each segment, the KNN process is called to find the split node for each segment. Zones are geographically connected using a metric based on the shortest path (Yan, H., & Weibel, R., 2008).

2 | VORONOI DIAGRAM

The Voronoi diagram shows the division of a certain area into several parts called cells, where each part contains a single point location (site). Each point in a cell has a closer distance to the site in that cell than other sites in the region. Thus, each point in the region has been paired with the closest site (Berg et al, 2008) in an area closer to the site corresponding to its region than other sites. The main concept of the voronoi diagram was discovered by Reitsma, R. and Trubin, S., (2007), "Given a number of distinct points in 2-D Euclidean space, a Voronoi diagram of a point set is a collection of regions that divides the plane and all locations in one region (except the region boundary) are closer to the corresponding point than to another point". While the mathematical concept was found by Novaes (2007): "Given a set of distinct points P = {P1, P2, …Pm} in a continuous space (plane), it will try to involve all other points of space with a closed set of m points. The closest member of the set P is the Voronoi diagram generator set, with m = 2".Let p = (p1, . . . , pn) and q = (q1, . . . , qn) be two points on Rn. The divisor of p and q consists of the points x = (x1, . . . , xn) where

$$||p - x|| = ||q - x|| \Longleftrightarrow ||p - x||2 = ||q - x||2 \Longleftrightarrow ||p||2 - ||q||2 = 2(p - q)^T x$$

Since p and q are different, this is a hyperplane equation, denoted by H(pi, pj) the closed half-space bounded by the bisector of pi and pj containing pi. In R2, H(pi, pj) is a half plane.The voronoi diagram shows the division of a certain area into several parts called cells, where each part contains a single point location called a site. Tiede, D. & Strobl, J., 2006). We define V(pi), the Voronoi cell for pi, as the set of points q that are closer to pi than to other sites. Thus, the voronoi cell for pi is defined by the equation:

$$V(pi) = \{q|\ distance\ (pi, q) < distance\ (pj, q, for\ j \neq 1\}\ or\ Pi = x|d(x, pi)\ d(x, pj)$$
$$V(pi) = \bigcap_{j \neq i} H(pi, pj)$$

3 | LOCATION POINT SEARCH BASED ON K-NN

By definition, a Voronoi diagram divides space into cells according to which site is closest. So, to determine the closest location, it is enough to compute a Voronoi diagram and generate a point location data structure for the Voronoi diagram. In this way, the nearest neighbor query is reduced to a point location query. It provides an optimal O(n) space and O(log n) query time method for answering queries for the location of points in the plane. Unfortunately, this solution does not generalize well to higher dimensions. The worst-case combinatorial complexity of a Voronoi diagram in the d dimension grows as O(n[d/2]) , and the optimal point location data structure is not known to exist in the higher dimensions. There are many ways to define the meaning of a mathematical equation as a model formula. Since the focus of this paper is on geometric approximations, we will assume that proximity is defined in terms of the known Euclidean distance. Most of the results that will be presented below can be generalized to the Minkowski metric, where the distance between two points p and q is defined in the equation below.

$$\mathrm{dist}_{m(p,q)} = \left(\sum_{k=i=1}^{d} |pi - qi|^2 \right)^{1/n}$$

The case m = 2 is the Euclidean distance, the case m = 1 is the Manhattan distance, and the limiting case m = 1 is the maximum distance. In typical geometric applications, the dimension d is assumed to be a constant constant and in complex or nongeometric metric spaces (P. Indyk, 2004). Of course, that without any preprocessing, the nearest neighbor finding problem cannot be solved in O(n) time via a simple brute-force search (Friedman et al., 2005).

3.1 | VCKNN Algorithm

In VCKNN as the development of the CKNN method, it seeks to eliminate weaknesses as a solution for finding location points, VCKNN based on voronoi diagrams also provides visibility of points of interest (POI) or which points of interest move out or enter the list and in which position the nodes will become separate nodes (split). nodes). The algorithm model for the VCKNN approach which is supported by the voronoi network diagram includes prepositions and proofs (Stergiopoulos et all, 2015). The preposition is the first step as generator of the Voronoi polygons that include the query point must be the nearest neighbor of the query point. While the proof is a statement in itself because the polygon defines the area where there is a point in the area that is closer to the polygon generator compared to other generators. The next step is the division of nodes (split nodes) in the Voronoi network diagram is determined by all the border points that intersect with the query path and the edges of the generator are split nodes. It is clear that, when the query path reaches the generator edge, the first NN will change because the distance to the shared edge generator has the same value.

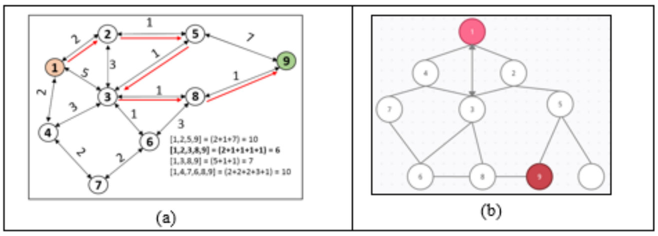

Fig. 2: *Dijkstra Tree (a) split nodes (b).*

3.2 | Kd-Tree Algorithm

For the class the most popular approach to k-nearest neighbor (k-NN) search would involve some sort of hierarchical spatial subdivision. Let S represent the set of n points on Rd whose question must be answered (Skiena, SS, 2008). In such an approach, the entire space is subdivided into successively smaller regions, and the resulting hierarchy is represented by a rooted tree (kd- Tree). Each tree node represents a region of space, called a cell. Implicitly, each node represents a subset of the S points located in its cell. The root of the tree is associated with the entire space and the entire set of points S. For some split nodes, if the number of points S associated with u is less than a constant, then this node is represented as a leaf from the tree. Otherwise, cells associated with u are recursively divided into smaller (possibly overlapping) sub cells according to some splitting rules. Then the corresponding points of S are distributed among these children according to which sub-cell they belong to.

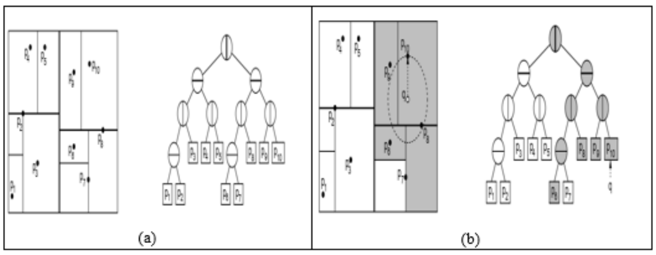

Fig. 3: *kd Tree from a set of vertices on from relatedspatial data (a) Search for nearest neighbors in kd Tree, shaded nodes visited(b).*

4 | RESULTS AND DISCUSSION

4.1 | The analysis carried out on the VCNN and KD-Tree algorithms

This is to use experimental data on the search from point q on P5 as a starting point and point x on P10 as a destination point as shown in Figure 4 is an illustration of an area that has various roads and health facilities in the form of puskesmas (Public health center) and referral hospitals. Finding the smallest number of distances by combining values that have alternative paths in the same direction as the destination . In Figure 4, point q is illustrated with node 1 while point x is illustrated with the purpose of finding a location.

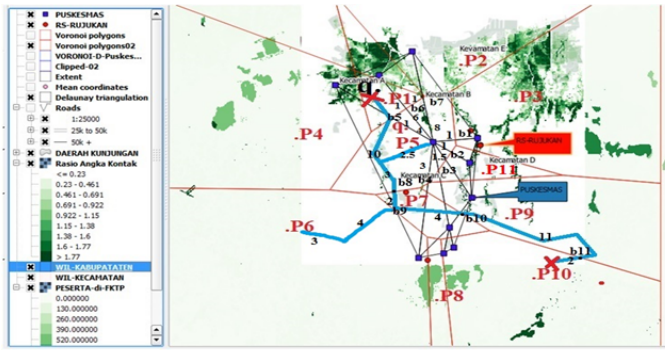

Fig. 4: *Application of VCKNN Based on Voronoi Diagrams*

This section not only describes the discussion of the VCKNN process, but also compares it with Kd-Trees in the application of voronoi diagrams in order to find location points. We will create a segmented calculation function and draw lines on the chart for easy understanding. Figure 4 shows an example of an experiment. The query is to find location points along the query path(P5 and P10), displayed as a thick blue line, starting at q at p5 and ending at P10. Border V (P5) i.e. b1, b2, b3, b4, b5, b6, b7 and the path from P1 to the border point is also shown. In the figure, it is found that the candidate interest points (CS) of polygon P5 are {P1, P2, P3,P4,P6,P7, P11}. With the discovery of the CS, a search is then carried out from the point of location q which is in V(P5) to the point of location on V(P10).

4.1.1 | VCKNN Algorithm Analysis

The first number of split nodes are intersection nodes between Voronoi polygons and moving paths as borders or borders (Solis, et all, 2009). In this case, split nodes = {b8, b9, b10, b11}. Split node b8 is the boundary point between Voronoi polygons V (P5) and V (P7), split node b9 is the boundary point between V (P6) and V (P7), and split node b10 is the border point between V (P7) and V (P9), and the split node b11 is the border point between V (P9) and V (P10). Here are the results of the identification of the NN group and through the split nodes V(P)

Table 1. Number of split nodes for each V

V(P)	Nearest Neighbor	Split Nodes
P5	{P1,P2,P3,P4,P6,P7,P11}	3
P7	{P5, P6, P8, P10, P9}	3
P9	{P5, P7, P10, P11}	2
P10	{P7,P8,P9}	1

Note that the results of the first NN are P5 with a distance range of 0.0 and 9.5, the second NN P7 with a distance range of 9.5 and 15.5, the third NN P9 with a distance range of 15.5 and 26.5, and the fourth NN p10 with a distance range of 26.5 and 28.5. In short, we can write the result of the first NN as a location search then the first NN as is {P5(0.0−9.5), P7(9.5−15.5), P9(15.5−26.5), P10(26.5−28.5)}. All distance ranges are distances from the initial query point. That is, when the query point q moves from 0 to 10, P5 is the first NN, and when q moves from 10 to 12, P7 will be the next NN, and so on until the location point P10 is found. Then for V (P5), V (P7), and V (P9), and V(P10) do the following steps. Take V(P5) as the first NN step:

Table 2. The movement of each point of the initial border location at P5

Movement distance q (km)	b1	b2	b3	b4	b5	b6	b7	b8
0.0	5.0	5.0	5.5	7.0	1.0	10	12	9.5
4.0	1.0	1.0	1.5	3.0	5.0	6.0	8.0	5.5
6.5	3.5	3.5	4.0	5.5	10	11	12	3.0
9.5	6.5	6.5	7.0	8.5	13	14	15	0

The path obtained in the search is q, (P5,P7,P9,P10) throughSplit Nodes = {b8, b9, b10, b11}. }. Split nodes are used only on map paths that have proximity to the next V(P). use all to compare to get the closest distance based on all segments.

4.1.2 | Algorithm Analysis KD-Tree

The first split nodes are all segments of the number of intersection paths in the Voronoi polygon and the paths move as borders or borders (Ricca et all, 2008). In this case, split nodes = {b1, b3,b2, b8, b9, b10, b11}. Split nodes b1 is the boundary point between V (P5) and V (P3), Split nodes b2 is the boundary point between V (P5) and V (P11), Split nodes b3 is the boundary point between V (P5) and V (P11) , Split node b8 is the boundary point between Voronoi polygons V (P5) and V (P7), split node b9 is the boundary point between V (P6) and V (P7), and split node b10 is the border point between V (P7) and V (P9), as well as the split node b11 are border points between V (P9) and V (P10). Here are the results of the identification of the NN group and through the split nodes V(P)

Table 3. Number of split nodes Each V

V(P)	Nearest Neighbor	Split Nodes
P3	{P2, P5, P11}	2
P5	{P1,P2,P3,P4,P6,P7,P11}	7
P7	{P5, P6, P8, P10, P9}	6
P9	{P5, P7, P10, P11}	7
P10	{P7,P8,P9}	4
P11	{P3, P5, P9}	3

Note that the results of the first NN are P3 with a distance range of 0.0 and 3.0, the second P5 with a distance range of 3.0 and 9.5, the third NN P7 with a distance range of 9.5 and 15.5, the fourth NN P9 with a distance range of 15.5 and 26.5, and the fifth NN P10 with a distance range 26.5 and 28.5, as well as the six P11s with a range of 28.5 and 31.5. In short, we can write the result of the first NN as a location search then the first NN as = {P3(0.0-3.0), P5(3.0−9.5), P7(9.5−15.5), P9(15.5−26.5), P10(26.5) 28.5), P11(28.5-31.5)}.

All distance ranges are distances from the initial query point. That is, when the query point q moves from 0 to 10, P5 is the first NN, and when q moves from 10 to 12, P7 will be the next NN, and so on until the location point P10 is found. Then for V (P5), V

(P7), and V (P9), and V(P10) do the following steps. Take V(P5) as the first NN step:

Table 4. The movement of each point of the initial border location at P5

Movement distance q (km)	b1	b2	b3	b4	b5	b6	b7	b8
0.0	5.0	5.0	5.5	7.0	1.0	10	12	9.5
2.5	2.0	3.0	4.5	2.0	10	8.0	6.5	60
4.0	1.0	1.0	1.5	3.0	5.0	6.0	8.0	5.5
6.5	3.5	3.5	4.0	5.5	10	11	12	3.0
9.5	6.5	6.5	7.0	8.5	13	14	15	0

The path obtained in the search is q, (P3, P5, P7, P9, P10, P11) throughSplit Nodes = {b1, b2,b3,b4,b5, b6, b7,b8, b9, b10, b11}. The split nodes that exist at the starting point are all used to compare to get the closest distance based on all segments.

4.1.3 | Evaluation and Comparison of Results (VKCNN and K-D Tree)

The evaluation carried out is to compare the VCKNN and K-DD Trees algorithms on aspects, number of split nodes, runtime and segmentation division by conducting experiments of 100 to 500 points of object search experimental locations in the voronoi diagram field. The data on the experimental digital map which has a road network and the location of the puskesmas and referral hospitals are combined with experimental data in the voronoi diagram field. The search between these two points is distinguished by the number of segmentations that are traversed starting from the query point on the voronoi shell diagram which is different from the point location of the object being searched for at random. Tables 6 and 7 below are the results of several experiments carried out on searches using the VCKNN and KD-Tree approaches which are used as references in making comparisons on the value of K = 5.

Table 5. Results of performance testing using the search method VCKNN with K = 5

Number of Location Points	Number of Split Nodes	Runtime (ms)	Average Segment Division
100	9	0.22	4
200	14	0.30	7
300	16	0.32	8
400	22	0.34	11
500	27	0.37	13

Table 6. Performance test results using the search method KD-Trees with K=5. Value

Number of Location Points	Number of Split Nodes	Runtime (ms)	Average Segment Division
100	29	0.90	6
200	31	0.93	8
300	34	0.95	9
400	36	0.96	12
500	37	0.97	15

5 | CONCLUSION

From the test results it can be concluded that the use of the VCKNN method can reduce the time required when searching for location points on a 2-dimensional plane (having a road network and object location points) such as location data on digital maps compared toKD-Tree. This means that the VCNN approach has the fastest response time in initializing the formation of a search route compared to KD-Tree. The time required for classification is also relatively constant even though the value of k varies. The search implementation with the VCNN approach also has a number of split nodes, from the test results obtained the time for initialization and the formation of the number of split nodes is directly proportional to the runtime. The fewer split nodes formed, the smaller the runtime. This research can be continued or developed by testing the time required for classification on the VCKNN and KD-Tree data structures.

BIBLIOGRAPHY

1. A. Okabe, B. Boots, K. Sugihara, and SN Chiu. “Spatial Tessellations, Concepts and Applications of VoronoiDiagrams”. John Wiley and Sons Ltd., 2nd edition, 2000.

2. Berg, Md, Cheong, O., Kreveld, Mv and Overmars, M., (2008), Computational Geometry: Algorithms and Applications. 3rd ed. Berlin: Springer.

3. H. Edelsbrunner. Algorithms in Combinatorial Geometry, volume 10 of EATCSMonographs on Theoretical Computer Science. Springer-Verlag, Heidelberg, West Germany, 1987.

4. GR Hjaltason and H. Samet. “Distance Browsing in Spatial Databases”. TODS, 24(2):265–318, 1999.

5. JH Friedman, F. Baskett, and LJ Shustek. An algorithm for finding nearest neighbors. IEEE Trans. Comput., C-24(10), 2005.

6. Novaes, AG (2007). Resolução de Problemas de Transporte com Diagramas de Voronoi, XXI ANPET, Panorama Nacional da Pesquisa em Transportes. Asociação Nacional de Pesquisa e Ensino em Transportes, Rio de Janeiro, Brazil.

7. P. Indyk. Nearest neighbors in high-dimensional spaces. In Jacob E. Goodman and Joseph O'Rourke, editors, Handbook of Discrete and Computational Geometry. CRCPress LLC, Boca Raton, FL, 2004. (To appear).

8. Reitsma, R. and Trubin, S., (2007), Information space partitioning using adaptive Voronoi diagrams. Information Visualization, 6, 123-138.

9. Ricca, F, Scozzari, A., and Simeone, B., (2008), Drawing political districts by weighted Voronoi regions and local search. Mathematical and Computer Modeling, 48, 1468-1477.

10. Solis, N., Rios-Mercado, RZ, and Alvarez, AM, (2009), Modelando sistemas territoriales con programacion entera. Ingenierias, 12 (44), 7-15.

11. Skiena, SS, (2008), The Algorithm Design Manual. 2nd ed. London: Springer.

12. Stergiopoulos, Y, Thanou, M, Tzes, (2015), A. Distributed collaborative coverage-control schemes for non-convex domains. IEEE Trans. auto. Control 2015, 60, 2422–2427.

13. Tiede, D. & Strobl, J., (2006), Polygon-based regionalization in a GIS environment [online]. In: E. Buhmann, S. Ervin, L. Jorgensen, and J. Strobl, eds. Trends in knowledge-based landscape modeling. Heidelberg: Wichmann, 54-59. Available from:http://www.masterla.de/ conf/ pdf/conf2006/ 23Tiede_L. pdf. [accessed May 02, 2019].

14. Yan, H., & Weibel, R., (2008), An algorithm for point cluster generalization based on the Voronoi diagram. Computers and GeoSciences, vol. 34, no. 8, pp. 939–954.

15. Yongxi, G., Guicai, L., Yuan, T., & Yaoyu, L., (2012), A vector-based algorithm to generate and update multiplicatively weighted Voronoi diagrams for points, polylines, and polygons. Computers & Geosciences, pp. 118-125.

How to cite this article: Sudaryanto ET AL.. Com parison of Finding the Location of the Near-est Health Facility Based On Knn-Voronoi Dia-gram. Research Review. 2022;597−603. https://doi.org/10.52868/RR/2022-3-1-3